The Dissertation Committee for Yong Min Wang
certifies that this is the approved version of the following dissertation:

A Column Generation Approach For Stochastic Optimization Problems

Committee:

Jonathan F. Bard, Supervisor

David P. Morton, Supervisor

Melba M. Crawford

Erhan Kutanoglu

Leon S. Lasdon

A Column Generation Approach For Stochastic Optimization Problems

by

Yong Min Wang, B.S., M.S.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2006

Dedicated to him who taught me truth.

Also dedicated to my wife Suk-Hwa who supports me as I am

and to those who have been praying for me.

Acknowledgments

I wish to thank my advisors Dr. Jonathan F. Bard and Dr. David P. Morton for their support and advice at each stage in the completion of this dissertation. It was my honor to work with them during my doctoral studies at the University of Texas at Austin.

Also I would like to thank the members of my dissertation committee: Drs. Melba Crawford, Erhan Kutanoglu and Leon Lasdon. Especially Dr. Crawford who gladly flew in from Indiana just for the defense and Dr. Kutanoglu who was able to arrive on time for the defense despite his need to accompany his wife to the hospital earlier in the morning. Dr. Lasdon's comments and suggestions regarding my future were greatly encouraging. I am indebted to all of their considered advise, their valued instruction, and their comments and corrections on my dissertation.

This dissertation would not have been possible without the sacrifice, unquestioning support and love of my beloved family – my wife Suk-Hwa, my son Joone and my daughter Eunjee. Their endless trust and unconditional love have been the sole source of my strength. It is my true feeling that this dissertation is due to the prayers of many people who have stood by me over the years.

A Column Generation Approach For Stochastic Optimization Problems

Publication No. ______________

Yong Min Wang, Ph.D.
The University of Texas at Austin, 2006

Supervisors: Jonathan F. Bard
David P. Morton

Understanding how uncertainty effects the dynamics and behavior of an organization is a critical aspect of system design. Models and methods that take uncertainty into account can lead to significant reductions in cost. This dissertation investigates the use of stochastic optimization models for the following applications: (1) a generalized assignment problem (GAP) with uncertain resource capacity and unknown processing times and (2) a shift planning and scheduling problem (SPSP) with unknown demand that arises in the construction of the workforce at US Postal Service mail processing and distribution centers.

In the models, the first stage decisions are made before the uncertainty is revealed. For the GAP, the first stage decisions correspond to an assignment of jobs to agents. Penalties are incurred when the assignments do not permit all demand to be satisfied. For the SPSP, the number of full-time and part-time

employees, as well as the number of full-time shifts by type, must be specified before the demand is known. In the second stage, feasibility is addressed by allocating overtime and calling in temporary workers to handle spikes in the mail volume.

This dissertation consists of three parts: (i) the development and analysis of stochastic integer programming models for the GAP and the SPSP, (ii) the estimation of the demand distributions from historical data for the Dallas processing and distribution center, and (iii) the development of column generation algorithms to solve the associated stochastic integer programs.

In the first part, the stochastic integer programming models are formulated for the two applications and the value of the stochastic solutions is presented. The second part begins with an analysis of the weekly demand and a test of hypothesis concerning the existence of an end-of-month effect, i.e., that the week at the end of the month might have larger volume. The hypothesis is rejected. After removing outliers, it is shown that the demand is approximated well by a normal distribution. In the last part of the dissertation, a branch-and-price algorithm is developed and used to solve the two applications. Experimental results demonstrate the algorithm's effectiveness.

Table of Contents

List of Tables

List of Figures

Chapter 1

Research Overview

1.1 Introduction

Mathematical programming has been used successfully to model and analyze a large variety of problems arising in industry. For purposes of planning and scheduling, it is common to collect historical data and fix parameter values based on average observations, hence ignoring the uncertainty. In many cases, this approach works well, especially when the corresponding deterministic models provide valuable insights and guidance. When uncertainty plays a significant role in the operations of a system, it is critical to account for it in the modeling process. The cost for doing so is often measured by a sharp increase in model complexity and computational effort.

Stochastic programming, first introduced in the 1950s, extends classical mathematical programming by considering uncertainty in the model [9, 20]. The two-stage stochastic program with recourse, which is one of the most common models, regards uncertainty as random variables with known probability distributions. In this context, there are at least three significant issues that must be addressed when using this technique: model formulation, estimating parameters, and developing an appropriate solution methodology.

This dissertation focuses on two applications of the two-stage stochastic model; (1) the generalized assignment problem (GAP) and (2) a shift planning and scheduling problem (SPSP). The GAP is standard combinatorial optimization problem that has a simple form. Martello and Toth [51] among others have shown it to be a NP-hard. The SPSP arises in the design of a permanent workforce at US Postal Service (USPS) mail processing and distribution centers, and is exceedingly more complicated. The goal of the dissertation is to formulate stochastic models for these applications and to develop efficient solution techniques that can be used for a broad class of similiar problems.

1.1.1 Generalized Assignment Problem

The generalized assignment problem (GAP) considers the minimum cost assignment of jobs to agents such that each job is assigned to one agent subject to capacity constraints. Despite its simple form, the deterministic GAP is a difficult combinatorial optimization problem that is NP-hard [52]. That said, the deterministic GAP has received considerable attention in the literature since being defined formally by Ross and Soland [57]. The virtue of the GAP varies from its direct application to it being a substructure of a more complicated model, such as a routing problem [27], a facility location model [58] or a computer networking application [4].

Recently researchers have extended GAP to capture uncertainty in the formulation, which is often present in real-world applications [1, 64, 66]. A stochastic GAP arises when we do not know an agent's resource capacity and/or the parameters for resource consumption. In this dissertation, we consider a stochastic GAP in which the coefficients for resource consumption and resource capacity are random, and a branch-and-price algorithm is proposed to solve the problem.

Most methods for solving the GAP exactly are based on branch-and-bound algorithms. Ross and Soland [57] propose a branch-and-bound algorithm for the GAP and Martello and Toth [51] and Fisher et al. [27] improve the branch-and-bound approach using different relaxation bounds. Martello and Toth [51] relax the equality constraints that ensure each job is done. Fisher et al. [27] relax these same constraints and tighten the corresponding bound using a dual-ascent procedure. Guignard and Rosenwein [33] extend the approach of Fisher et al. [27] by allowing temporary primal infeasibility and by adding surrogate constraints whenever the solution violates primal feasibility. They point out that their algorithm's performance degrades as the ratio of the number of agents to the number of jobs increases.

Often, exact solutions are not necessary and near optimal solutions that can be obtained quickly are sufficient. Some approximation algorithms are based on the linear programming (LP) relaxations. For example, Trick [67] develops a heuristic algorithm which repeatedly solves the LP relaxation and fixes variables taking integer values, either with probability one or some probability less than one. Narciso and Lorena [54] propose an approximation algorithm using the Lagrangian relaxation described above, and a subgradient algorithm for improving the multipliers.

Stochastic variants of the GAP have received much less attention in the literature than the deterministic GAP. In the stochastic model, job-agent assignments are made in the first stage. After the resource-consumption parameters are realized, recourse is handled by reassignments. Albareda-Sambola et al. [1] introduce a stochastic GAP in which resource consumption is modeled as a Bernoulli random variable to capture yes-no demand for individual jobs. By assuming the resource-consumption parameters are independent of the agent, Albareda-Sambola et al. [1] show that the combinatorial recourse problem is totally-unimodular. They exploit this property in solving the problem with the L-shaped decomposition method with binary first stage variables and continuous second stage variables. (The L-shaped method [70] is a cutting-plane method similar to Benders' decomposition.)

Tokas et al. [66] develop stochastic versions of the multi-resource GAP, which is introduced by Gavish and Pirkul [28]. In Tokas et al. [66], the resource-consumption coefficients are deterministic but the agents' resource capacity is random. Recourse in [66] is handled differently on three models. The capacity constraints are handled by: (i) penalizing the magnitude of infeasibilities, (ii) penalizing the number of violated resource constraints or (iii) canceling jobs. Normal, bimodal of two normal distributions and exponentially-distributed resource-capacity distributions are considered. Tokas et al. [66] iteratively tighten Lagrangian relaxation bounds and use branch-and-bound to solve these models.

Spoerl and Wood [64] propose a stochastic model of the GAP, where the resource-consumption parameters are normally distributed and the other parameters are deterministic. They construct two deterministic equivalent formulations, depending on whether an agent's resource-consumption parameters have a common mean-to-variance ratio.

In this dissertation, we consider the GAP where the resource-consumption and capacity parameters are uncertain, which has not been studied in the literature and a column generation approach is applied to solve the problem.

1.1.2 Shift Planning and Scheduling Problem

The design of a permanent workforce requires a balance between full-time, part-time, and temporary employees. In manufacturing, this is a relatively easy problem to solve, especially when demand is steady and each classified employee works 5 days in a row, 8 hours per day. In that case, a straightforward cost analysis is often all that is needed. However, when demand fluctuates from one day to the next and when daily operations span more than a standard shift, the solution is much less obvious. Trying to find the size and composition of a permanent workforce to minimize the annual cost of running a facility requires the consideration of a host of factors including overtime rules, the use of temporary labor, government regulations, union contracts, and organizational polices. Part-timers, for example, can increase management's options for dealing with peaks in demand but their use may be limited by agreements negotiated with labor unions. Full-timers are more stable and reliable but impose greater costs. Temporary labor is the least expensive option but those who are available may not have the necessary skills or motivation for the job.

Partly for these reasons, improved shift scheduling plays a central role in many organizations wishing to streamline their workforce. Some of the more familiar examples include nurse scheduling in hospitals, crew scheduling in the airline industry, staffing of call centers, and manpower planning at mail processing facilities. Both the long-run and midterm problems faced by organizations in these industries are further complicated by uncertainty in demand, lack of good data, and the inability to hire and fire workers at will due to training requirements, skill shortages, and binding job security agreements.

In the long run, the goal of companies and organizations driven by service is to optimize the regular workforce by finding weekly schedules or tours for each employee in a given skill category. This means specifying the work days, their length, the daily start times, and the lunch breaks. At mail processing and distribution centers, for example, these specifications define a "bid job" [37].

The first component of tour construction has traditionally involved shift scheduling to satisfy some percentage of the average daily demand. The corresponding objective is to find the optimal crew size and daily job assignment for each member of the crew, each day of the week [2]. A shift may vary in length depending on whether it is associated with a full-time or part-time employee, and may extend into the following day.

Personnel scheduling has been widely studied for more than 50 years beginning with the work of Dantzig [19] who developed a set covering formulation for the problem. Different ways of dealing with labor requirements, such as days off and breaks, are discussed in [2, 3, 11, 16]. Various problem definitions associated with types of workers, working hour limitation, and skill categories can be found in [2, 10, 11].

Burns and Carter [16] investigated the problem of finding the minimum workforce size subject to various days off constraints. For the case in which each employee must be given a predefined number of days off per week in addition to A out of every B weekends off, they derived lower bounds on the workforce size and proved that the bounds were exact. Bechtold and Jacobs [11] dealt with the issue of assigning lunch breaks to shifts by developing an implicit formulation of the assignment problem. They derived a series of forward and backward constraints that guaranteed feasibility and then used a post-processor to make the actual assignments. Testing showed that their implicit formulation was superior to the traditional set-covering formulation in terms of computational time and memory usages.

In an early application, Segal [62] solved a shift scheduling problem for telephone operators who require breaks and reliefs using various network flow formulations. Computation times were around 30 seconds on IBM360/67 for problems in which the day was divided into 15 minute periods from 9 am to 6 pm.

Brusco and Jacobs [15] addressed restricted start-time constraints for a tour scheduling problem arising at United Airlines airport ground stations and proposed a two-stage heuristic solution strategy. An instance with 2000 variables was solved on a Pentium-based (100 MHz) PC and shown to outperform the methods being used at the time. A set of experiments was conducted and an analysis of variance (ANOVA) was used to confirm the superiority of the proposed algorithm.

Bard (2004) considered uncertainty in both demand and employee availability by increasing the size of the permanent workforce and decreasing the number of working days allocated to part-timers. An essential component of the work was a procedure for identifying the week to be used to define the input data. The algorithm in the paper considered historical productivity and absenteeism rates, union rules regarding casual labor, and internal limits on the use of overtime. Results were presented for the Dallas mail processing and distribution centers based on data from fiscal years 1999-2001.

Lau [45] defined a variation of a shift scheduling problem that arises in airline crew scheduling, which he called the *changing shift assignment problem* (CSAP). The problem is concerned with finding a satisfactory assignment of shifts to workers subject to demand and shift change constraints. Using transformation from 3-satisfiability, he showed that the decision version of CSAP is NP-hard. When the shift change constraints are monotonic, though, he was able to develop a polynomially solvable algorithm. Computational results for random instances involving 40 workers, 7 periods, and 3 shifts were presented.

In this dissertation, the focus is on the combined tour and shift scheduling problem at USPS mail processing and distribution centers (P&DC). Rather than using some average level of demand to fix the permanent workforce, however, we intend to treat demand as a random variable and develop a stochastic optimization approach that minimizes the expected annual cost of running a P&DC.

1.1.3 Column Generation

The two-stage stochastic programming problem with recourse becomes more difficult to solve as the number of scenarios grows. This difficulty is compounded by the fact that many specialized algorithms are not suitable for problems with integer variables. In most scheduling problems, integrality is a requirement, at least for a subset of the variables. Recently, column generation schemes have been proposed to handle this case.

Dantzig and Wolfe [21] decomposition exploits specially-structured subproblems to solve linear programs using a reformulation with exponentially-many columns. More generally, a column-generation method (CG) deals with a mathematical program with an enormous number of columns. The columns are iteratively generated by solving a subproblem to identify columns with attractive reduced costs. In some integer programs (like the GAP), CG is applied to a problem with exponentially-many columns because more compact formulations have weaker LP relaxations. In other problems, a formulation with a huge number of columns is the only model available. Column generation can be applied at the root node of a branch-and-bound algorithm for the purpose of tightening the initial LP relaxation bound. When the pricing subproblem can be extended to apply to nodes throughout the branch-and-bound tree the method is called a branch-and-price procedure (B&P). In recent decades, B&P has been used with considerable success (e.g., [8, 72]) on a variety of problems including the GAP [61], facility location problems [63], vehicle routing problems [24], and scheduling problems [7, 23].

Effective CG depends on being able to solve the pricing subproblems quickly and on the algorithm converging in a reasonable number of iterations. Since one of its early applications to a cutting-stock problem [30], CG has had a reputation of converging slowly. In order to improve its convergence, stabilization methods have been developed. (See Lübbecke and Desrosiers [47] for a survey.) The pricing subproblems are formed using dual variables from the master problem, which contains columns generated so far, and attempts to "stabilize" dual solutions from the master problem limit the distance between the dual solutions from one iteration to the next [50]. This can be accomplished by slightly relaxing constraints and penalizing "infeasibilities." This limits the dual variables, and these limits and associated penalties can be updated dynamically [25]. Other approaches resolve the problem among multiple-optimal dual solutions by using an interior point [59] and use a weighted average of previously generated solutions and the current dual solution [73]. We employ the method described above of du Merle et al. [25].

1.2 Research Objectives

The primary objective of this research has been to investigate the generalized assignment problem and the USPS shift planning and scheduling problem from a two-stage stochastic integer programming point of view. In so doing, a number of stochastic models have been developed and analyzed. Solution methods have also developed and their performance evaluated through an extensive experimental design. To solve many of the large-scale integer linear programs that arose in the research, it was necessary to develop a column generation scheme. Little success was achieved by simply calling on a commercial code.

Integral to the work was the estimation of the distribution of the demand data for the USPS problem. These data were obtained from the Dallas P&DC and represent three years of weekly mail volume. For planning purposes, it was assumed that mail volume is proportional to demand.

To speed convergence of the column generation scheme, an comprehensive investigation of several “stabilization” methods was undertaken. The intent of these schemes, in general, is to improve the quality of the dual solutions of the master problem.

1.3 Outline of the Dissertation

Chapter 2 contains a brief review of stochastic programming and the development of stochastic models. Section 2.1 describes stochastic programming and the meaning of solutions obtained from the various formulations. The development of stochastic models and computational results for the generalized assignment problem follow in Section 2.2. A stochastic version of the shift planning and scheduling problem that addresses uncertain demand is provided in Section 2.3. In Section 2.4, demand data from the Dallas P&DC is analyzed, and a distribution is estimated. A statistical analysis is included to verify the results. Section 2.5 includes the computational results and the development of an approximation algorithm.

In Chapter 3, column generation methods are described. Section 3.1 explains column generation and the mechanics of stabilization. A small computational experiment is carried out for a facility location problem to demonstrate how stabilization can improve algorithmic performance. In Section 3.2, a column generation formulation is given for the generalized assignment problem and branching strategies are to be examined within the context of branch and price. Finally, in Section 3.3, the shift planning and scheduling problem is reformulated using a column generation approach.

Chapter 4 concludes the dissertation by summarizing its contributions and presenting some ideas for future research.

Chapter 2

Stochastic Programming Models

This chapter reviews the literature and concepts of stochastic programming and develops the stochastic formulations for the GAP and SPSP. The demand distribution for the USPS application is also estimated from historical data.

2.1 Stochastic Programming Review

Stochastic optimization methods are called for when some parameters of a model are uncertain but can be characterized by a known probability distribution. With a steady increase in computer power, it is now possible to develop and study stochastic extensions of deterministic models that were heretofore out of reach due to their size. Applications that fall into this category include the unit commitment problem in electrical power generation [22, 32, 65], capacity planning [26, 36] and revenue management [53, 71]. In these applications, the most common random variables are demand and capacities.

Beale [9] and Dantzig [20] were the first to introduce linear programming under uncertainty. Since then, two-stage stochastic programming with recourse has received a significant amount of attention, e.g., see [14, 40, 55].

Stochastic linear programs (SLPs) with recourse that have a small number of scenarios may be solved as large-scale linear programs via the simplex method or an interior point method. As the number of scenarios grows algorithms that exploit special structure are computationally preferable. Approaches to solve such problems include cut generation algorithms and decomposition methods—by scenario and by stage.

The L-shaped algorithm [70], a cut-generation algorithm, may be viewed as applying Benders' decomposition [12, 29] in a way that decomposes the problem with respect to the possible outcomes of the random variables or *scenarios*.

In decomposition methods, some constraints are relaxed and satisfied iteratively. Relaxing nonanticipativity produces subproblems associated with scenarios and relaxing linking constraints leads to subproblems associated with stages. Scenario decomposition is very effective for problems that have a specialized structure, such as a network, or for problems whose single scenario subproblems that can be solved efficiently (e.g., see [14, 34, 40, 55]) Decomposition by stage, which is sometimes referred as nodal decomposition, does not remove the stochastic complication in the problem because the nonanticipativity constraint is still present. Rosa and Ruszczyński [56] applied this approach to a multistage problem.

In a two-stage stochastic model, the second-stage cost known as the recourse function, is represented as a function of the first stage decision variables. For problems with linear recourse, the recourse function is piecewise linear and convex.

If integer restrictions are imposed on the first-stage variables only, existing technology described above carries over fairly easily, at least in principle. However, if integer restrictions are imposed on some or all of the second-stage decision variables, which is generally the case in scheduling, convexity, and even continuity, of the recourse function are lost. Therefore, much of the research on stochastic integer programs (SIPs) has concentrated on finding a good approximation (or a good bound) and developing algorithms to solve ever larger problems in reasonable time.

Convex approximations of SIPs with simple integer recourse are applicable for problems in which random right-hand-side parameters follow a certain class of distributions [42, 68]. Work to extend this to include problems with more general integer recourse is ongoing [46, 69].

Wollmer [74] solved models with 0-1 first stage variables and continuous second-stage problems, and Laprote and Louveaux [43] extended this to models with binary first-state variables and an arbitrary second-stage problem. Carøe and Tind [18] proposed an extension of the L-shaped method to solve SIPs that required a branch-and-bound algorithm for the second-stage integer problems. However, since the cuts generated during the branching process are not convex in general, more research needs to be done to apply their approach to general integer problems.

For applications, Laprote et al. [44] solved a class of capacitated facility location problems with stochastic demand by formulating them as stochastic integer programs with binary variables in the first stage and continuous variables in the second stage. They reported computational results for randomly generated problem instances with up to 40 customers and 10 facility locations.

Lulli and Sen [49] developed a B&P algorithm for a multistage mixed integer program by relaxing the nonanticipativity constraints. They solved stochastic lot sizing problems having up to six decision stages and about 400 integer variables in 48 minutes on a SUN Ultra 80. CPLEX was used as the optimization engine, with the tolerance set at 0.0001. However, they implemented the column generation method using an open source code called BCP, part of COIN-OR [35]. Because BCP took a significant amount of time to traverse the search tree they concluded that a more efficient procedure was needed if the approach was to be useful.

Lagrangian relaxation was used by Gollmer et al. [32], Takriti et al. [65] to solve the unit commitment problem with uncertain demand. In this problem, it is necessary to determine the operating schedule (binary variables) of the generators over a planning horizon that could span a month at a time. Dentcheva et al. [22] used the bundle method with CPLEX for the subgradient algorithm and a specialized algorithm to solve the subproblems. They presented computational results for a problem instance having 20 thermal units, 6 hydro plants and 192 time periods. Solutions were obtained in less than two minutes (CPU-time) on an HP-Apollo 735/125 with an optimality gap of 1.5% in worst case. Carøe and Schultz [17] used dual-decomposition with Lagrangian relaxation to solve randomized test problems having 10 binary variables and 65 continuous variables. The solutions were within 0.2% of the optimum and computational times were less than 10 minutes on a DEC Alpha workstation with a 266 MHz processor. The algorithm was implemented with CPLEX and NOA [41].

2.1.1 Stochastic Linear Programming

In a two-stage stochastic linear program (SLP), a decision x must be made before the uncertain parameters are realized. A second set of decisions y is then taken as recourse actions once the uncertainty has been revealed. The goal is to make the here-and-now decision x, that minimizes the total expected cost.

Let $\tilde{q}, \tilde{D}, \tilde{B}$ and $\tilde{d}$ be random variables and $\tilde{\xi} : \Xi \rightarrow \Re^n$ be a vector formed by the components of $\tilde{q}, \tilde{D}, \tilde{B}$ and $\tilde{d}$ where Ξ is the support of $\tilde{\xi}$ and n is the number of random parameters in $\tilde{q}, \tilde{D}, \tilde{B}$ and $\tilde{d}$. Then the SLP can be formulated as follows:

$$z = \min_{x} \{cx + E_{\tilde{\xi}}[h(x, \tilde{\xi})] : Ax \geq b, \, x \in X\} \tag{2.1}$$

where $E_{\tilde{\xi}}$ is the mathematical expectation with respect to $\tilde{\xi}$ and X contains simple bounds. The recourse function $h(x, \tilde{\xi})$ is defined as

$$h(x, \tilde{\xi}) = \min_{y} \{\tilde{q}y : \tilde{D}y \geq \tilde{B}x + \tilde{d}, \, y \in Y\} \tag{2.2}$$

where Y is a set of simple bounds. We call x the first-stage decision vector, the set $\mathcal{X} = \{x : Ax \geq b, x \in X\}$ is the first stage constraint set, and (2.2) the second stage problem, in which the first stage decisions x are constant. Let ξ^ω be a realization of $\tilde{\xi}, \omega \in \Omega$ where Ω is the sample space of $\tilde{\xi}$, with $P(\tilde{\xi} = \xi^\omega) = p^\omega$. Then the deterministic equivalent program (DEP) can be defined as follows.

$$\begin{aligned} z = \min_{x,y} \quad & cx + \sum_{\omega \in \Omega} p^\omega q^\omega y \\ \text{s.t.} \quad & Ax \geq b \\ & B^\omega x + D^\omega y \geq d^\omega \\ & x \in X, \quad y \in Y \end{aligned}$$

Definition 2.1 (Relatively Complete Recourse)**.** *Let $\mathcal{K} = \{x : \exists\, y$ satisfying $\tilde{D}y \geq \tilde{B}x + \tilde{d}$, wp 1$\}$. If $\mathcal{X} \cap \mathcal{K} = \mathcal{X}$, then the SLP is said to have relatively complete recourse.*

Relatively complete recourse means that we can find a feasible solution to the second stage problem for any first stage solution x. We assume that problem (2.1) has relatively complete recourse.

Proposition 2.1. *$h(x, \tilde{\xi})$ is convex in x for a fixed value of $\tilde{\xi} \in \Xi$.*

Proposition 2.2. *$h(x, \tilde{\xi})$ is convex on Ξ for fixed $x \in \mathcal{X} \cap \mathcal{K}$.*

For the proofs and additional properties of $h(x, \tilde{\xi})$, see, e.g., Kall and Wallace [40]. Using these properties, the SLP can be solved by cutting plane methods such as the L-shaped method [14, 70].

If uncertainty is not revealed at once, the model (2.1) is extended to a multistage problem. At time $t, t = 1, ..., |T|$, we decide the decision (x_t) with the information revealed so far (ξ_{t-1}) and the previous stage decision (x_{t-1}).

$$
\begin{aligned}
z = \min_{x_0} \quad & c_0 x_0 + E_{\tilde{\xi}_1}[h_1(x_0, \xi_1^\omega)] \\
\text{s.t.} \quad & A_0 x_0 \geq b_0, \\
& x_0 \in X_0
\end{aligned}
\tag{2.3}
$$

where

$$
\begin{aligned}
h_t(x_{t-1}, \tilde{\xi}_t) = \min_{x_t} \quad & \tilde{c}_t x_t + E_{\tilde{\xi}_{t+1}}[h_{t+1}(x_t, \tilde{\xi}_{t+1})] \\
\text{s.t.} \quad & \tilde{A}_t x_t \geq \tilde{B}_t x_{t-1} + \tilde{b}_t \\
& x_t \in X_t
\end{aligned}
\tag{2.4}
$$

with $h_{T+1} \equiv 0$.

2.1.2 Stochastic Integer Programming

Stochastic Integer Programming (SIP) can also be represented as (2.1) and (2.2) except that X and/or Y has integer restrictions. If integer restrictions are imposed on the first stage variables, then the objective function of the problem has the same properties as in SLP since the recourse function $h(x, \tilde{\xi})$ is not changed. We can use the same solution techniques for SLP coupled with some branch-and-bound methods to obtain an integer first stage solution. However, integer restrictions on the second stage variables changes the problem considerably. The objective function, in general, is no longer convex.

As a result, research interests in SIP have focused on the case that the second stage problem requires integral decisions.

Proposition 2.3. *If the SIP has relative complete recourse, then the recourse function $h(x, \tilde{\xi})$ is lower semicontinuous in x for all $\tilde{\xi} \in \Xi$.*

Proof. See Birge and Louveaux [14], p.124. □

Proposition 2.4. *The recourse function may be discontinuous.*

2.1.3 Value of the Stochastic Solution

Stochastic programs enable us to capture uncertainty using random variables in mathematical programming models, but the computational burden may be high. Therefore, it is important to measure the value of stochastic programming solutions over simpler and less computationally expensive ways of handling uncertainty. One possibility is to replace the random variables by their expected values and solve the associated deterministic problem. Another is to solve several deterministic problems, each representing a scenario and then to extract a decision from the separated scenario solutions. Applying heuristics may provide reasonable solutions. In this section, two ways of measuring the quality of stochastic solutions and relevant bounds are discussed.

To see the value of stochastic solution, several relevant problems are solved. Using the same notation as in Section 2.1.1, define

$$RP = \min_{x \in \mathcal{X}} \{cx + E_{\tilde{\xi}} h(x, \tilde{\xi})\} \tag{2.5}$$

$$EV = \min_{x \in \mathcal{X}} \{cx + h(x, E_{\tilde{\xi}}[\tilde{\xi}])\} \tag{2.6}$$

$$WS = E_{\tilde{\xi}} \min_{x \in \mathcal{X}} \{cx + h(x, \tilde{\xi})\} \tag{2.7}$$

where $\tilde{\xi}$ denotes the random variable, $\mathcal{X}$ is the feasible set for x, and $h(x, \tilde{\xi})$ is the optimal value of the second stage problem as defined in Section 2.1.1. RP stands for recourse problem, in which expected cost is minimized, and EV is the expected value problem, which is the one-scenario problem defined by the mean of ξ. WS denotes the wait-and-see problem, which is the weighted sum of the optimized one-scenario problems associated with each scenario. Let x_{EV} denote the solution to the EV problem. Define

$$EEV = cx_{EV} + E_{\tilde{b}} h(x_{EV}, \tilde{b}), \tag{2.8}$$

where EEV denotes the expected cost of using x_{EV}. Decision x_{EV} is feasible but, in general suboptimal for (2.5), hence $EEV \geq RP$. In (2.7) we know perfect information regarding the future and so $RP \geq WS$.

The value of the stochastic solution VSS is defined as

$$VSS = EEV - RP, \tag{2.9}$$

and denotes the cost of using the suboptimal decision x_{EV} decision to the solution to (2.5). The expected value of perfect information ($EVPI$) is the difference between the wait-and-see and the here-and-now solution:

$$EVPI = RP - WS, \tag{2.10}$$

and denotes the value we would be willing to pay to eliminate uncertainty.

2.2 Generalized Assignment Problem

The deterministic GAP has been studied and applied considerably in many application, even though the problem is known as NP-hard, due to the fact that the problem appears to be useful as a substructure in many applications, such as vehicle routing problem, as well as its own benefits. Recently many researchers consider uncertainty in different parameters of the problem which is often the case in real-world application.

In this study, uncertainty is allowed for the capacity of resources and the consumption of each agent's capacity for each job. This problem can be formulated as a two stage stochastic problem, in which the assignment decision should be made only with the knowledge of the distribution of uncertainty and once the uncertainty is realized, the recourse decision is to made. The objective is to minimize the total expected cost.

In this section, the generic model will be introduced and the column generation formulation will follow in the next chapter as a branch-and-price algorithm. The performance of the algorithm will be reviewed with computational results.

2.2.1 Formulation

The generalized assignment problem (GAP) considers the minimum cost assignment of jobs to agents such that each job is assigned to one agent subject to capacity constraints. The following notation is used to formulate the problem.

Sets

I set of agents; $i = 1, ..., m$

J set of jobs; $j = 1, ..., n$

Parameters

c_{ij} assignment cost of job j to agent i

q_i cost for an additional unit of capacity of agent i over the given amount

d_{ij} consumption of capacity for agent i for job j

b_i capacity of agent i

Decision variables

x_{ij} 1 if job j is assigned to agent i, 0 otherwise

The problem can be presented as follows.

$$\min \quad \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} \tag{2.11a}$$

$$\text{s.t.} \quad \sum_{i \in I} x_{ij} = 1, \quad \forall\, j \in J \tag{2.11b}$$

$$\sum_{j \in J} d_{ij} x_{ij} \le b_i, \quad \forall\, i \in I \tag{2.11c}$$

$$x_{ij} \in \{0, 1\}, \quad \forall\, i \in I,\, j \in J \tag{2.11d}$$

The objective function (2.11a) minimizes the total assignment cost and the constraint (2.11b) enforces that each job is assigned to one agent. The constraint (2.11c) ensures that the capacity of agent i is obeyed and finally the integrality is maintained by (2.11d).

Stochastic Model Now we consider uncertainty in the resource-capacity and resource-consumption parameters. Let $\tilde{\xi}_i = \left((\tilde{d}_{ij})_{j\in J}, \tilde{b}_i\right)$ denote the random vector of the parameters associated with agent i, and let $\xi_i^\omega = \left((d_{ij}^\omega)_{j\in J}, b_i^\omega\right)$ be its realizations indexed over the sample space $\omega \in \Omega_i$. We assume $|\Omega_i|$ is finite and let $p_i^\omega = P(\tilde{\xi}_i = \xi_i^\omega)$, $\omega \in \Omega_i$, be the corresponding probability mass function. The vector of all the random parameters is $\tilde{\xi} = (\tilde{\xi}_i)_{i\in I}$, but as we describe below, our model's objective function is separable in the subvectors $\tilde{\xi}_i$, $i \in I$, and hence the dependency structure among these subvectors is irrelevant. Restated, an optimal solution to our model is optimal for all dependency structures between these subvectors.

Like all the stochastic GAP models [1, 64, 66] discussed in the previous section, our stochastic GAP model first assigns jobs to agents subject to constraints (2.11b) and (2.11d) and with costs as indicated in (2.11a). Then we observe a realization of $\tilde{\xi}$ and penalize the sum of the magnitudes of violations in constraints (2.11c) over all agents, with respective unit penalties $q_i \geq 0$. We add the expected value of this penalty function,

$$\begin{aligned} E_{\tilde{\xi}} \sum_{i\in I} q_i \left(\sum_{j\in J} \tilde{d}_{ij} x_{ij} - \tilde{b}_i \right)^+ &= \sum_{i\in I} q_i E_{\tilde{\xi}_i} \left(\sum_{j\in J} \tilde{d}_{ij} x_{ij} - \tilde{b}_i \right)^+ \\ &= \sum_{i\in I} q_i \sum_{\omega\in\Omega_i} p_i^\omega \left(\sum_{j\in J} d_{ij}^\omega x_{ij} - b_i^\omega \right)^+, \quad (2.12) \end{aligned}$$

to that in (2.11a). Here, $(\cdot)^+ = \max(\cdot, 0)$. Linearizing the positive-part terms with continuous variables y_i^ω, denoting the magnitude of the constraint violation, we have

$$
\begin{aligned}
z^* = \min_{x,y} \quad & \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} + \sum_{i \in I} q_i \sum_{\omega \in \Omega_i} p_i^\omega y_i^\omega && \text{(2.13a)} \\
\text{s.t.} \quad & \sum_{i \in I} x_{ij} = 1, \quad j \in J && \text{(2.13b)} \\
& \sum_{j \in J} d_{ij}^\omega x_{ij} - y_i^\omega \le b_i^\omega, \quad \omega \in \Omega_i,\ i \in I && \text{(2.13c)} \\
& x_{ij} \in \{0, 1\}, \quad i \in I,\ j \in J && \text{(2.13d)} \\
& y_i^\omega \ge 0, \quad \omega \in \Omega_i,\ i \in I. && \text{(2.13e)}
\end{aligned}
$$

The objective function in (2.13a) minimizes the job assignment costs as in (2.11a) plus the expected cost of violating the agent-capacity constraints. Constraints (2.13c) and (2.13e) linearize the positive-part terms in (2.12), while constraints, (2.13b) and (2.13d) are identical to (2.11b) and (2.11d).

2.2.2 Computational Results

In this section, the value of stochastic solution (VSS) is investigated for various problems. The deterministic equivalent model (2.13) was implemented in JAVA with CPLEX 9.0 and solved on a PC with a dual 1.8GHz CPU, 1GB memory, running SuSE Linux. For the integer solutions, the relative tolerance was set with the default value 0.0001 and two-hour time limit was applied.

The algorithms were tested on four classes of random problems, which were based on the rules for deterministic instances organized by Martello and Toth [51]. Let $m = |I|$ and $n = |J|$.

A. c_{ij} and d_{ij} are integer from a uniform distribution between 10 and 25 and between 5 and 25, respectively. $b_i = 9\frac{n}{m} + 0.4 \max_{1 \le i \le m} \sum_{j \in J^*} d_{ij}$ where $J_i^* = \{j | i = \arg\min_{1 \le r \le m} c_{rj}\}$. $q_i = 30$ for all i.

B. Same as A for c_{ij}, d_{ij} and q_i. $b_i = 0.7$ of b_i in A.

C. Same as A for c_{ij}, d_{ij} and q_i. $b_i = \frac{0.8}{m} \sum_{j \in J} d_{ij}$.

D. Same as C for b_i. d_{ij} is integer from a uniform distribution between 1 and 100. $c_{ij} = 100 - d_{ij} + k$, where k is integer from a uniform distribution between 1 and 21. $q_i = 125$ for all i.

For all four classes, scenarios for d_{ij}^{ω} were generated from a uniform distribution between $(d_{ij} - 20\%)$ and $(d_{ij} + 20\%)$ and similarly, b_i^{ω} were generated from a uniform distribution between $(b_i - 20\%)$ and $(b_i + 20\%)$. Various sizes were used ($m = 5, 10, 20$ and $n = 30, 50$).

Table 2.1 shows the objective function values and the values of stochastic solutions. The name of each problem is composed of a letter and three numbers. The letter stands for the problem class and the numbers tell the numbers of agents, jobs and scenarios in turn. For all problems, VSSs were quite high. For an example, A.5.30.10 had 46.56% of VSS, which means that we can decrease the total expected cost by 46.56% by solving the stochastic model rather than solving the deterministic problem using mean values. Suppose we solve the problem with mean values of the random variables and wait for the realization of random parameters. If an assignment under a scenario is over the capacity, the additional payment has to be paid and hence the total expected cost would be increased. For A.5.30.10 case, handling uncertainty in this way cost 46.56% more than solving the stochastic model. Note that some RP values were the same as EV values, i.e., for the cases of A.20.50.10 and B.20.50.10, which means that we can handle the uncertainty with the same expected cost as the EV problem which would result in 78% and 43% higher expected costs (EEV values), respectively, when the random variables are realized.

2.3 Shift Planning And Scheduling Problem

The problem that we consider involves the development of a set of bid jobs and schedules for a permanent workforce of career employees comprised of full-time regulars (FTRs), part-time regulars (PTRs), and part-time flexibles (PTFs). We begin with a presentation of the model in which the size and composition of the workforce selected must be sufficient to meet all the demand for some representative week during the year, where deterministic demand values are specified in 1/2-hour increments by labor category over the selected week. In this deterministic version of the problem, overtime and the use of casual labor is not considered. Then, we extend the model to handle random demand.

2.3.1 Deterministic Problem

We consider a long-term planning problem at USPS P&DCs in which we want to decide the number of workers and to assign them to specific shifts.

Table 2.1: Results for various problems

Problems	VSS (%)	Objective value EV	EEV	RP
A. 5.30.10	46.56	362.0	532.0	363.0
A. 5.50.10	93.04	617.0	1196.8	620.0
A.10.30.10	111.26	324.0	690.8	327.0
A.10.50.10	28.35	554.0	714.9	557.0
A.20.30.10	12.12	308.0	346.5	309.0
A.20.50.10	78.48	513.0	915.6	513.0
B. 5.30.10	37.04	331.0	456.3	333.0
B. 5.50.10	40.23	606.0	856.8	611.0
B.10.30.10	6.21	325.0	348.4	328.0
B.10.50.10	49.59	551.0	824.2	551.0
B.20.30.10	16.55	305.0	357.8	307.0
B.20.50.10	43.62	516.0	741.1	516.0
C. 5.30.10	5.42	384.0	414.0	392.7
C. 5.50.10	15.23	591.9	690.2	599.0
C.10.30.10	35.29	349.0	488.4	361.0
C.10.50.10	15.14	571.0	662.3	575.2
C.20.30.10	32.64	322.5	458.0	345.3
C.20.50.10	46.62	535.4	880.0	600.2
D. 5.30.10	9.53	3062.3	3366.0	3073.0
D. 5.50.10	16.85	5045.0	5916.0	5063.0
D.10.30.10	38.32	3010.9	4201.0	3037.1
D.10.50.10	34.05	4960.5	6688.4	4989.6
D.20.30.10	26.46	2992.2	3814.1	3016.0
D.20.50.10	36.63	4939.7	6865.3	4973.0

Table 2.2: An example set of shifts

Worker type	Starting time	Shift lengths (periods)	Number of shifts
FTR	Every hour	9	n^T
PTF	Every hour	4, 5, 6, 7	4 n^T

We use various types of workers, each having different lengths of work periods, and different restrictions. The framework of the weekly scheduling model is adopted from the work by Bard et al. [6], which was built in consultation with the Austin P&DC. In this problem, the 'best' week is selected and a weekly schedule is formed with this 'best' week's demand as proposed by Bard [5]. This weekly schedule is to be used repeatedly for the long-term planning period.

In this model, each day is divided into n^T periods. Of the three types of career employees at P&DCs, only FTRs and PTFs are considered in this dissertation; the USPS is currently trying to phaseout the few PTRs on the payroll. An FTR works 8 hours a day and is given a 1-hour lunch break. A PTF can work one of several different shift lengths and is given a 1-hour lunch break if the shift is longer than 6 hours. This break must occur during a predefined break window. A shift may start at the beginning of any period in the day, so the total number of shifts is n^T for FTRs and n^T times the number of different shift lengths for PTFs. An example set of shifts is shown in Table 2.2.

Depending on the policy of the P&DC, an FTR must start the same time each day or may be given a *starting time window*. In the latter case, the employee is assigned to one of the shifts within the time window which typically spans two hours. PTFs have a starting time window of six hours, which means, for example, that a part-timer may have a 4-hour shift on Monday that starts at 6:00 a.m., and a 7-hour shift on Wednesday that starts at 11:00 a.m. FTRs must have two days off per week. The total number of PTFs is limited by the union contract such that the ratio of full-time to part-time workers ρ can be maintained. Also a PTF is not allowed to work more than $\bar{H}$ hours per week on average.

The following notation is used in the development of the model.

Indices

- d index for the days of the week; $d = 1, ..., 7$
- t, s index for the periods during a day; $t = 1, ..., n^T$
- f index for the full-time shift type; $f = 1, ..., n^F$
- p index for the part-time shift type; $p = 1, ..., n^P$
- a index for the full-time starting window; $a = 1, ..., n^A$
- b index for the part-time starting window; $b = 1, ..., n^B$

Parameters

- c_a^F weekly cost of a full-time worker with starting time window a
- c_p^P cost for a part-time worker assigned to shift p

k earliest period a break can begin for any shift

q latest period a break can begin for any shift

n^T number of time periods in a day

n^F number of full-time shifts

n^P number of part-time shifts

n^A number of starting time windows for full-time workers

n^B number of starting time windows for part-time workers

ρ full-time to part-time labor ratio

$\bar{\theta}$ upper bound on average working hours per week for a part-time worker

D_{td} demand for period t on day d (man-hours)

Incidence matrices

A_{af} 1 if full-time shift f belongs to the starting time window a; 0 otherwise

B_{bp} 1 if part-time shift p belongs to the starting time window b; 0 otherwise

B^F_{sf} 1 if full-time shift f's break window lies between period s and n^T; 0 otherwise

$\bar{B}^F_{sf}$ 1 if full-time shift f's break window lies between period s and n^T of the next day; 0 otherwise

B^P_{sp} 1 if part-time shift p's break window lies between period s and n^T; 0 otherwise

$\bar{B}^P_{sp}$ 1 if part-time shift p's break window lies between period s and n^T of the next day; 0 otherwise

F^F_{sf} 1 if full-time shift f's break window lies between period 1 and s; 0 otherwise

$\bar{F}^F_{sf}$ 1 if full-time shift f's break window lies between period 1 and s of the next day; 0 otherwise

F^P_{sp} 1 if part-time shift p's break window lies between period 1 and s; 0 otherwise

$\bar{F}^P_{sp}$ 1 if part-time shift p's break window lies between period 1 and s of the next day; 0 otherwise

G_{ft} 1 if full-time shift f covers period t; 0 otherwise

$\bar{G}_{ft}$ 1 if full-time shift f covers period t of the next day; 0 otherwise

P_{pt} 1 if part-time shift p covers period t; 0 otherwise

$\bar{P}_{pt}$ 1 if part-time shift p covers period t of the next day; 0 otherwise

S_f 1 if full-time shift f's break is in the day the shift starts; 0 otherwise

$\bar{S}_f$ 1 if full-time shift f's break is in the day after the shift starts; 0 otherwise

T_p 1 if part-time shift p has a break in the day the shift starts; 0 otherwise

$\bar{T}_p$ 1 if part-time shift p has a break in the day after the shift starts; 0 otherwise

Decision variables

x_{fd} number of full-time workers assigned to shift f on day d

y_{pd} number of part-time workers assigned to shift p on day d

β_{td} total number of breaks in period t on day d

w_a number of full-time workers with starting time window a

v_b number of part-time workers with starting time window b

Model

$$\min_{w,v,x,y,\beta} \quad \sum_{a=1}^{n^A} c_a^F w_a + \sum_{d=1}^{7} \sum_{p=1}^{n^P} c_p^P y_{pd} \tag{2.14a}$$

$$\text{s.t.} \quad \sum_{f=1}^{n^F} (G_{ft} x_{fd} + \bar{G}_{ft} x_{f,d-1}) + \sum_{p=1}^{n^P} (P_{pt} y_{pd} + \bar{P}_{pt} y_{p,d-1}) - \beta_{td} \geq D_{td}, \ \forall t, d, \tag{2.14b}$$

$$\sum_{a=1}^{n^A} w_a - \rho \sum_{b=1}^{n^B} v_b \geq 0, \tag{2.14c}$$

$$w_a - \frac{1}{5} \sum_{d=1}^{7} \sum_{f=1}^{n^F} A_{af} x_{fd} \geq 0, \quad \forall a, \tag{2.14d}$$

$$w_a - \sum_{f=1}^{n^F} A_{af} x_{fd} \geq 0, \quad \forall a, d, \tag{2.14e}$$

$$v_b - \frac{1}{5}\sum_{d=1}^{7}\sum_{p=1}^{n^P} B_{bp} y_{pd} \geq 0, \quad \forall b, \tag{2.14f}$$

$$v_b - \sum_{p=1}^{n^P} B_{bp} y_{pd} \geq 0, \quad \forall b, d, \tag{2.14g}$$

$$\bar{\theta}\sum_{b=1}^{n^B} v_b - \sum_{d=1}^{7}\sum_{t=1}^{n^T}\sum_{p=1}^{n^P} P_{pt} y_{pd} \geq 0, \tag{2.14h}$$

$$\sum_{t=k}^{s} \beta_{td} - \sum_{f=1}^{n^F}(F^F_{sf} x_{fd} + \bar{F}^F_{sf} x_{f,d-1}) - \sum_{p=1}^{n^P}(F^P_{sp} y_{pd} + \bar{F}^P_{sp} y_{p,d-1}) \geq 0, \; \forall s, d, \tag{2.14i}$$

$$\sum_{t=s}^{q} \beta_{td} - \sum_{f=1}^{n^F}(B^F_{sf} x_{fd} + \bar{B}^F_{sf} x_{f,d-1}) - \sum_{p=1}^{n^P}(B^P_{sp} y_{pd} + \bar{B}^P_{sp} y_{p,d-1}) \geq 0, \; \forall s, d, \tag{2.14j}$$

$$\sum_{t=1}^{T} \beta_{td} - \sum_{f=1}^{n^F}(S_f x_{fd} + \bar{S}_f x_{f,d-1}) - \sum_{p=1}^{n^P}(T_p y_{pd} + \bar{T}_p y_{pd}) = 0, \quad \forall d, \tag{2.14k}$$

$$w_a,\, v_b,\, x_{fd},\, y_{pd},\, \beta_{td} \in Z_+, \quad \forall\ a,\, b,\, f,\, p,\, t,\, d \tag{2.14l}$$

The objective function (2.14a) minimizes the total weekly cost of the workforce. The first component is written in terms of the number of full-time workers for each time window a while the second component is written in terms of the number of part-time shifts.

The demand at each time period on each day is satisfied by (2.14b). Incidence matrices are used to identify the number of FTRs and PTFs that are available at time t on day d. The break variable β_{td} accounts for those who are at lunch during their shift. The second constraint, (2.14c), referred to as the ratio constraint, limits the number of PTFs. In general, it costs less to use the part-time employees than to use the full-timers because of the differences in benefits. If no limit is imposed on the number of PTFs, the demand would be satisfied only by PTFs. Almost every union contract limits the work by the part-timers in various ways and USPS uses head counts on the payroll as an upper bound.

Constraints (2.14d)-(2.14g) are upper bounds of the numbers of workers, introduced by Burns and Carter [16] in a slightly different form. The first of these bounds in (2.14d) ensures that there is enough coverage so that every worker can take two days off a week; e.g., the total number of FTR shifts over all types f in time window a used over the week must be no more than 5 times the number of workers w_a enrolled in starting time window a. Constraint (2.14f) provides the bound associated with PTFs. The second sets of these bounds, (2.14e) and (2.14g), are needed to assure that there are a sufficient number of workers, FTRs and PTFs, respectively, for each starting time window to cover the day with the highest demand. Constraint (2.14e) is the result of linearizing the following:

$$w_a \geq \max \left\{ \sum_{f=1}^{n^F} x_{fd} : f \text{ is in time window } a,\ d = 1, ..., 7 \right\}$$

where x_{fd} is the number of FTRs assigned to shift type f on day d. Similarly, constraint (2.14g) is for PTFs. At optimality, one of these bounds is tight. The restriction on the average working hours for PTFs is maintained by (2.14h).

To take account of breaks, we use three constraints (2.14i)-(2.14k), which were introduced by Bechtold and Jacobs [11]. The first constraint (2.14i) is referred to as the *forward pass* constraint, which assures that the total number of breaks initiated from the starting period of the break window, k, up to a given period s exceeds the total number of employees who should have taken their breaks by that period so that sufficient breaks are provided. The employees included in the constraint are only those whose break windows are fully covered through s. The second constraint (2.14j) is referred to as the *backward pass* constraint and assures that the total number of breaks that are initiated from a given period s through the last period of the break window, q, exceeds the number of employees who are entitled to a break during this interval. In other words, there should be sufficient breaks in the future from s onward to satisfy the break requirement for the rest of the day. These two constraints are needed to provide every employee with a break, but they are not sufficient to enforce the requirement that exactly one break be assigned to each worker entitled to one. Furthermore, they do not limit the break assignments to their respective ranges. The constraint, (2.14k), which is known as the *balance equation*, is needed to assure that every worker is assigned a break and that it is within its permitted window. For details about break constraints, see Bard et al. [6] and Bechtold and Jacobs [11].

The solution to model (2.14) provides the number of breaks that are required in each period over the week but does not associate them with shifts. The actual assignment of breaks can be made with a post-processing procedure that solves a transportation problem in which the supply nodes correspond to the the breaks and the demand nodes correspond to the shifts; however, because there is no objective function, a greedy algorithm that quickly finds feasible solutions is all that is needed. Finally, constraint (2.14l) states that all variables must be integral.

2.3.2 Stochastic Formulation

In general the demand, D_{td}, is not known when we must make certain scheduling decisions. In the model of Section 2.3.1 we use some point forecast of the demands, using, for example, a choice of the "best week". Here, we relax this assumption and replace the point forecast with an estimate of the probability distribution governing demand and formulate an associate stochastic program with the hope that resulting schedules will better hedge against uncertainties in future demand.

We consider a planning horizon of one year. The deterministic model produces a weekly schedule, which is applied repeatedly for that year. We must decide the number of full-time and part-time workers that will form the workforce before the demand is known. This is generally the case in a scheduling situation because the regular workforce must be hired in advance and its size cannot be changed easily. Hence, in the stochastic programming model we decide workforce size in the first stage, before realizing the demand, and fix it after the demand is revealed in the second stage. If demand cannot be met using the existing workforce with regular-time hours then we can, with some restrictions, assign overtime and hire so-called casual workers. Casual workers have the same shifts as part-time workers but are temporary in that they are called in as needed, and work for a day.

With respect to our modeling constructs from Section 2.3.1, the number of full-time workers for each time window a (w_a), the number of part-time workers for each time window b (v_b), and the number of full-time workers for each shift f (x_{fd}) must be decided before the demand is realized. After demand is realized part-time workers are assigned to each shift p (y_{pd}). In addition, overtime hours may be assigned to full-time workers either as two-hour or four-hour extensions of their existing shifts via new decision variables μ^1_{fd} and μ^2_{fd}. The number of casual workers for each shift p (γ_{pd}) is selected. The goal is to satisfy demand through this four types of assignments, but restrictions on the use of casual workers, the way in which overtime can be assigned, etc., sometimes make this impossible. Decision variables w^U_a and x^U_{fd} account for unmet demand. Casual workers do not have breaks.

Union agreements with the USPS limits overtime hours (H_{OT}) no more than 6% of the total hours ($H_A + H_{OT}$) worked by full-time and part-time workers, where H_A denotes the regular-time hours worked by full-time and part-time workers. Thus,

$$\frac{H_A}{H_{OT} + H_A} \leq 0.06$$

which yields

$$\frac{H_{OT}}{H_A} \leq \frac{0.06}{0.94} = 0.0625.$$

That is, the allowable fraction of regular-time hours which can be de devoted to overtime hours is $\rho^{OT} = 0.0625$.

The USPS wage rates increase by a factor of 1.5 for the first two hours of overtime work and by a factor of 2 for hours in excess of two hours. Working more than four hours of overtime is not allowed. (See Figure 2.1). The ratio of casual workers to regular workers may not exceed ρ^C; the USPS uses $\rho^C = 0.059$.

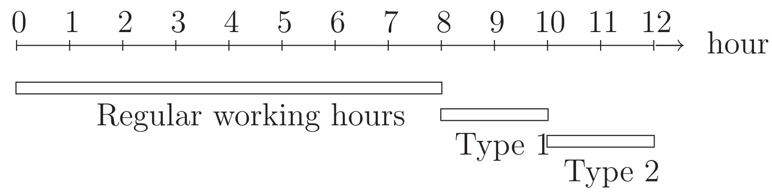


Figure 2.1: Two types of overtime

Multi-Stage vs. Two-Stage Problem A 52-week version of our problem can be formulated as a multistage stochastic problem: At the beginning of the one-year planning period, before any random demands become known, the size of permanent workforce is determined; denote this sizing decision $\hat{x}_0$. Then, in subsequent weeks of the year, $\tau = 1, ..., 52$, we observe the demand for the week and construct an adaptive weekly schedule using overtime for the existing full-time employees as well as scheduling part-time employees and casual workers. Denote the schedule for week τ by $\hat{x}_\tau$. The dimension of $\hat{x}_\tau$, $\tau = 0, ..., 52$ is denoted $\hat{n}^\tau$. Then the problem may be stated

$$\begin{aligned} \min_{\hat{x}_0} \quad & c_0 \hat{x}_0 + E_{\tilde{b}_1}[h_1(\hat{x}_0, \tilde{b}_1)] \\ \text{s.t.} \quad & A_0 \hat{x}_0 = b_0 \\ & \hat{x}_0 \in Z_+^{\hat{n}_0}, \end{aligned} \tag{2.15}$$

where for $\tau = 1, ..., 52$

$$\begin{aligned} [h_\tau(\hat{x}_0, \tilde{b}_\tau)|(\tilde{b}_1, ..., \tilde{b}_{\tau-1})] = \min_{\hat{x}_\tau} \quad & c_\tau \hat{x}_\tau + E_{\tilde{b}_{\tau+1}|\tilde{b}_1,...,\tilde{b}_\tau}[h_{\tau+1}(\hat{x}_0, \tilde{b}_{\tau+1})|(\tilde{b}_1, ..., \tilde{b}_\tau)] \\ \text{s.t.} \quad & A_\tau \hat{x}_\tau = \tilde{b}_\tau - B_\tau \hat{x}_0 \\ & \hat{x}_\tau \in Z_+^{n_\tau} \end{aligned} \tag{2.16}$$

with $h_{53} \equiv 0$ and $\tilde{b}_\tau, \tau = 1, ..., 52$, including random demand. The constraint set in (2.15) restricts the workforce-sizing decision and the feasible region of (2.16) constrains schedules of the overtime, part-time workers and the casual workers, given the sizing decision $\hat{x}_0$.

Note that the second term in the objective function of (2.16) does not depend on $\hat{x}_\tau$ and is simply a constant with respect to the optimization in (2.16). Moreover, the optimal solution $\hat{x}_\tau^*$ to (2.16) depends only on the demand $\tilde{b}_\tau$ and not on the previous demand $\tilde{b}_1, ..., \tilde{b}_{\tau-1}$, i.e., $\hat{x}_\tau^* = \hat{x}_\tau^*(\tilde{b}_\tau)$. Thus, the expected cost incurred in week τ only depends on the marginal distribution of $\tilde{b}_\tau$ and the initial workforce sizing decision. As a result, our multi-stage problem (2.15)-(2.16) reduces to the following two-stage SIP.

$$\begin{aligned} \min_{\hat{x}_0} \quad & c_0 \hat{x}_0 + \sum_{\tau=1}^{52} E_{\tilde{b}_\tau} \left[h_\tau(\hat{x}_0, \tilde{b}_\tau) \right] \\ \text{s.t.} \quad & A_0 \hat{x}_0 = b_0 \\ & \hat{x}_0 \in Z_+^{\hat{n}_0} \end{aligned}$$

where, for $\tau = 1, ..., 52$,

$$
\begin{aligned}
h_\tau(\hat{x}_0, \tilde{b}_\tau) = \min_{\hat{x}_\tau} \quad & c_\tau \hat{x}_\tau \\
\text{s.t.} \quad & A_\tau \hat{x}_\tau = \tilde{b}_\tau - B_\tau \hat{x}_0 \\
& \hat{x}_\tau \in Z_+^{\hat{n}_\tau}
\end{aligned}
\tag{2.17}
$$

If each of the $\tilde{b}_\tau$, $\tau = 1, ..., 52$, have the same distribution, then we can simplify (2.17) by putting $\tilde{b} \leftarrow \tilde{b}_\tau$, $A \leftarrow A_\tau$, $B \leftarrow B_\tau$ and $c \leftarrow c_\tau$, $\tau = 1, ..., 52$. This leads to

$$
\sum_{\tau=1}^{52} E_{\tilde{b}_\tau} \left[h_\tau(\hat{x}_0, \tilde{b}_\tau) \right] = 52 E_{\tilde{b}} \left[h(\hat{x}_0, \tilde{b}) \right], \tag{2.18}
$$

where $h(\hat{x}_0, \tilde{b})$ is the equivalent of (2.17) with the subscript τ removed.

Two-Stage Model Now we define notation that will be used for the detailed parameters and variables in addition to those defined in Section 2.3.1.

Parameters

c_p^P cost of part-time worker assigned to shift p

c_f^{OT1} cost of type 1 overtime for a full-time worker assigned to shift f

c_f^{OT2} cost of type 2 overtime for a full-time worker assigned to shift f

C_p^C cost of casual worker assigned to shift p

C^U cost per week for additional full-time workers needed to cover unmet demand

ρ^{OT} maximum allowable ratio of overtime hours to regular-time hours

$\underline{\rho}^{OT}$ lower bound on ratio of overtime hours that must be achieved before casual workers can be used

ρ^{C} maximum allowable ratio of casual workers to permanent workers

$\underline{\theta}$ lower bound on average working hours per week that must be assigned to part-time workers before casual workers can be used

Incident matrices

G_{ft}^{OT1} 1 if type 1 overtime on shift f covers period t; 0 otherwise

G_{ft}^{OT2} 1 if type 2 overtime on shift f covers period t; 0 otherwise

Random parameters

$\widetilde{D}_{td}$ demand for period t on day d

First stage decision variables

w_a number of full-time workers with starting time window a

v_b number of part-time workers with starting time window b

x_{fd} number of workers assigned to full-time shift f on day d

Second stage decision variables

y_{pd} number of part-time workers assigned to shift p on day d

β_{td} total number of breaks in period t on day d

μ^1_{fd} number of full-time workers assigned to type 1 overtime on shift f on day d

μ^2_{fd} number of full-time workers assigned to type 2 overtime on shift f on day d

γ_{pd} number of casual workers assigned to shift p on day d

w^U_a number of additional full-time workers in starting time window a needed to cover unmet demand

x^U_{fd} number of additional full-time workers on shift f on day d needed to cover unmet demand

z binary variable used to enforce logical constraints regarding the precedence order in which overtime hours, part-time worker hours, and casuals can be assigned

With regard to demand, let D^{ω}_{td} be a realization of $\widetilde{D}_{td}, \omega \in \Omega$, where Ω represents the sample space and ω is a sample point. We assume $\widetilde{D}_{td}$ has a discrete distribution with a finite number of scenarios. The corresponding probability mass function is given by $P(\widetilde{D}_{td} = D^{\omega}_{td}) = p^{\omega}_{td}$.

In the deterministic model, incidence matrices with "bars" are used, e.g., $\bar{G}_{ft}$ and $\bar{P}_{pt}$ in (2.14b), because shifts may continue to the next day. However, we write the stochastic model as if all shifts are contained within a single day, which allows us to drops those terms that have "bars". We do this only to simplify the presentation. Our computational work is based on the full model.

Formulation

$$\min_{w,v,x} \quad 52\left(\sum_{a=1}^{n^A} c_a^F w_a + E_{\widetilde{D}}\left[h(v_b, x_{fd}, \widetilde{D})\right]\right) \tag{2.19a}$$

$$\text{s.t.} \quad \sum_{a=1}^{n^A} w_a - \rho \sum_{b=1}^{n^B} v_b \geq 0, \tag{2.19b}$$

$$w_a - \frac{1}{5}\sum_{d=1}^{7}\sum_{f=1}^{n^F} A_{af} x_{fd} \geq 0, \quad \forall a, \tag{2.19c}$$

$$w_a - \sum_{f=1}^{n^F} A_{af} x_{fd} \geq 0, \quad \forall a, d, \tag{2.19d}$$

$$w_a, v_b, x_{fd} \in Z_+ \quad \forall a, b, f, d, \tag{2.19e}$$

where $h(w_a, v_b, x_{fd}, D^\omega) =$

$$\min_{y,\beta,\mu^1,\mu^2,\gamma,w^U,x^U} \quad \sum_{d=1}^{7}\sum_{p=1}^{n^P} c_p^P y_{pd} + \sum_{d=1}^{7}\sum_{f=1}^{n^F}(c_f^{OT1}\mu_{fd}^1 + c_f^{OT2}\mu_{fd}^2) + \sum_{d=1}^{7}\sum_{p=1}^{n^P} c_p^C \gamma_{pd} + \sum_{a=1}^{n^A} c^U w_a^U \tag{2.20a}$$

$$\text{s.t.} \quad \sum_{p=1}^{n^P} P_{pt} y_{pd} + \sum_{f=1}^{n^F}(G_{ft}^{OT1}\mu_{fd}^1 + G_{ft}^{OT2}\mu_{fd}^2) + \sum_{p=1}^{n^P} P_{pt}\gamma_{pd} - \beta_{td} + \sum_{f=1}^{n^F} G_{ft} x_{fd}^U \geq D_{td}^\omega - \sum_{f=1}^{n^F} G_{ft} x_{fd}, \quad \forall\ t,\ d \tag{2.20b}$$

$$\frac{1}{5}\sum_{d=1}^{7}\sum_{p=1}^{n^P} B_{bp} y_{pd} \leq v_b, \quad \forall\ b \tag{2.20c}$$

$$\sum_{p=1}^{n^P} B_{bp} y_{pd} \leq v_b, \quad \forall\ b,\ d \tag{2.20d}$$

$$\sum_{d=1}^{7}\sum_{t=1}^{n^T}\sum_{p=1}^{n^P} P_{pt} y_{pd} \leq \bar{\theta}\sum_{b=1}^{n^B} v_b \tag{2.20e}$$

$$\sum_{t=k}^{s}\beta_{td} - \sum_{p=1}^{n^P} F_{sp}^P y_{pd} \geq \sum_{f=1}^{n^F} F_{sf}^F x_{fd}, \quad \forall\ s,\ d \tag{2.20f}$$

$$\sum_{t=s}^{q}\beta_{td} - \sum_{p=1}^{n^P} B_{sp}^P y_{pd} \geq \sum_{f=1}^{n^F} B_{sf}^F x_{fd}, \quad \forall\ s,\ d \tag{2.20g}$$

$$\sum_{t=1}^{n^T}\beta_{td} - \sum_{p=1}^{n^P} T_p y_{pd} = \sum_{f=1}^{n^F} S_f x_{fd}, \quad \forall\ d \tag{2.20h}$$

$$\sum_{d=1}^{7}\sum_{t=1}^{n^T}\sum_{f=1}^{n^F}(G_{ft}^{OT1}\mu_{fd}^1 + G_{ft}^{OT2}\mu_{fd}^2) - \rho^{OT}\sum_{d=1}^{7}\sum_{t=1}^{n^T}\sum_{p=1}^{n^P} P_{pt} y_{pd} \leq \rho^{OT}\sum_{d=1}^{7}\sum_{t=1}^{n^T}\sum_{f=1}^{n^F} G_{ft} x_{fd} \tag{2.20i}$$

$$\mu^1_{fd} \leq x_{fd}, \quad \forall\ f,\ d \tag{2.20j}$$

$$\mu^2_{fd} - \mu^1_{fd} \leq 0, \quad \forall\ f,\ d \tag{2.20k}$$

$$\sum_{d=1}^{7}\sum_{t=1}^{n^T}\sum_{p=1}^{n^P} P_{pt}\gamma_{pd} - \rho^C \sum_{d=1}^{7}\sum_{t=1}^{n^T}\sum_{p=1}^{n^P} P_{pt}y_{pd} \leq \rho^C \sum_{d=1}^{7}\sum_{t=1}^{n^T}\sum_{f=1}^{n^F} G_{ft}x_{fd} \tag{2.20l}$$

$$\sum_{d=1}^{7}\sum_{t=1}^{n^T}\sum_{p=1}^{n^P} P_{pt}y_{pd} \geq \underline{\theta}\sum_{b=1}^{n^B} v_b - M(1-z) \tag{2.20m}$$

$$\sum_{d=1}^{7}\sum_{t=1}^{n^T}\sum_{f=1}^{n^F} (G^{OT1}_{ft}\mu^1_{fd} + G^{OT2}_{ft}\mu^2_{fd}) - \underline{\rho}^{OT} \sum_{d=1}^{7}\sum_{t=1}^{n^T}\sum_{p=1}^{n^P} P_{pt}y_{pd} + M(1-z)$$

$$\geq \underline{\rho}^{OT} \sum_{d=1}^{7}\sum_{t=1}^{n^T}\sum_{f=1}^{n^F} G_{ft}x_{fd} \tag{2.20n}$$

$$\gamma_{pd} \leq Mz, \quad \forall\, p,\ d, \tag{2.20o}$$

$$w^U_a - \frac{1}{5}\sum_{d=1}^{7}\sum_{f=1}^{n^F} A_{af}x^U_{fd} \geq 0, \quad \forall\ a \tag{2.20p}$$

$$w^U_a - \sum_{f=1}^{n^F} A_{af}x^U_{fd} \geq 0, \quad \forall\, a,\ d \tag{2.20q}$$

$$z \in \{0, 1\},$$

$$y_{pd},\ \beta_{td},\ \mu^1_{fd},\ \mu^2_{fd},\ \gamma_{pd},\ x^U_{fd},\ w^U_a \in Z_+, \quad \forall\ f,\ p,\ t,\ d,\ a \tag{2.20r}$$

The first stage feasible region is defined by the full-time to part-time ratio constraint (2.19b) and the two lower bound constraints (2.19c) and (2.19d) on the number of FTRs. The objective function (2.19a) minimizes the *yearly* workforce costs which include the expected second stage cost. Constraints (2.19b)-(2.19d) are identical to those used in the deterministic model.

In the second stage problem, the objective function (2.20a) accounts for the costs associated with the variable component of the workforce. Included are costs for PTFs, type 1 and type 2 overtime, casual workers, and unmet demand, respectively. Constraint (2.20b) ensures that as much of the (net) demand as possible is covered by the available workforce. Because the number of FTRs (w_a) and their bid job assignments (x_{fd}) are fixed in the first stage, the demand is reduced on the right-hand side of (2.20b) in each period of the day. Unmet demand is accounted for by the variable w_a^U, which is penalized in the objective function to discourage its use.

As in the deterministic model, (2.20c) and (2.20d) provide bounds on the part-time employee assignments while (2.20e) limits their total working hours per week. Constraints (2.20f)-(2.20h) account for breaks, overtime hours are limited by (2.20i), and the number of casual hours is limited by (2.20l). Constraint (2.20j) ensures that the number of type 1 overtime shifts associated with full-time shift f doesn't exceed the number of FTRs assigned to that shift. A similar restriction is imposed by constraint (2.20k) for type 2 overtime.

USPS policy is to use as much overtime and part-time hours as possible before hiring casuals who are actually paid at the lowest rate due to the absence of benefits. Therefore, the number of casual shifts should be zero unless a substantial portion of the available part-time and overtime hours are assigned. Constraints (2.20m)-(2.20o) implement this restriction via the binary variable z; the value of the constant M is chosen to be sufficiently large. Constraints (2.20m) and (2.20o) ensure that the minimum number of part-time hours ($\underline{\theta}$)

is assigned before casuals are hired while constraint (2.20n) has the same effect for overtime hours. When $z = 0$, no casual shifts are permitted by (2.20o) and constraints (2.20m) and (2.20n) are redundant. When $z = 1$, (2.20m) and (2.20n) ensure that both part-time and overtime hours are within the ranges that permit hiring of casuals.

Constraints (2.20p) and (2.20q) are used to account for unmet demand. Because all demand must be covered in practice, we have introduced a second category of FTRs as a modeling device, but associate a higher than normal cost with their use. The variable x^U_{fd} indicates how many additional full-time shifts are needed and w^U_a indicates the corresponding number of FTRs.

As can be seen from the derivation in the previous section, an interesting property of model (2.19) is that there is no requirement that the demand be uncorrelated from week to week. For equation (2.18) and hence this model to be correct then, all that is needed is for each week's demand to have the same (marginal) distribution.

2.4 Distributions of Demand for US Postal Service

Model (2.19)-(2.20) is driven by the demand for labor, specified in man-hours for each time period in the week. Even in the deterministic case, however, these data are difficult to obtain because only aggregate records exist. The only practical way is to translate the aggregated weekly mail volume into hourly labor requirements with a model. Therefore, once the weekly data are analyzed, we will use daily mail arrival profiles for an average week in conjunction with the equipment scheduling algorithm developed by Zhang [75], to derive values for D_{td}. Note that the weekly demand is given in units of volume of mail pieces while the daily demand is specified by the labor-hours required to process the associated mail.

In this section, the distribution of the weekly demand will be estimated using historical data, and given a realization of weekly demand the daily demand will then be generated from the mail arrival profile generated by Zhang's algorithm. More specifically his procedure is used to turn a scaled demand for one week, in terms of mail volume, into labor-hour demand per hour for the seven days of the week.

Table 2.3 shows the weekly volume of mail (in 1000s of pieces) for the Dallas P&DC for fiscal years 1999-2001. As can be seen from the table, the weeks 13 to 16 differ from the rest of the year, due to the Christmas holiday season. To prevent distortion, these weeks are excluded from our analysis.

2.4.1 Weekly Demand Data

The US federal fiscal year starts on October 1, and the data we have begins on October 5 1998.

When one considers the nature of mail delivery in the US, it is natural to suspect an end-of-month (EOM) effect, where the weeks at the end of each month have higher volume than the others due to, say, bill payments and invoicing. If this is the case, it would be necessary to separate the distribution of the end of the month from the 'regular' weekly demand distribution.

Table 2.3: Weekly mail volume for Dallas P&DC from 1999 to 2001 (1000s of pieces)

Weeks	1999	2000	2001
1	108,220.20	107,355.80	106,310.00
2	100,163.20	105,947.50	107,773.50
3	105,290.00	110,358.40	106,875.40
4	113,982.90	117,655.70	113,758.60
5	106,055.90	108,914.90	109,334.20
6	106,681.70	111,553.10	111,913.30
7	105,550.00	105,850.80	107,135.60
8	109,262.80	116,047.70	114,570.50
9	100,785.90	101,010.10	104,664.70
10	105,806.20	107,565.50	116,640.30
11	97,720.70	91,674.70	93,753.30
12	109,050.00	107,628.90	102,962.90
13	119,479.30	117,379.30	124,625.50
14	124,167.40	121,949.90	116,373.40
15	86,571.30	102,926.70	124,409.70
16	89,890.80	85,295.70	82,400.00
17	104,751.70	121,554.50	99,631.70
18	113,673.30	112,445.50	116,416.90
19	108,996.50	102,791.40	110,484.00
20	119,134.10	107,624.40	113,178.50
21	115,284.70	117,333.60	117,451.50
22	106,715.00	107,082.80	113,260.70
23	96,705.40	103,882.80	109,012.00
24	102,258.80	100,287.20	101,427.80
25	111,712.10	107,091.70	110,501.70
26	104,991.30	107,936.50	112,565.10
27	103,446.30	104,468.40	107,338.00
28	103,803.60	103,872.40	103,069.00
29	104,821.10	99,248.30	108,015.20
30	107,266.30	109,026.00	112,189.40
31	102,831.90	105,378.90	108,316.60
32	99,662.50	104,497.90	104,454.10

Table 2.3: Weekly mail volume for Dallas P&DC from 1999 to 2001 (cont'd)

33	101,046.50	102,678.60	106,920.90
34	115,380.50	116,801.50	114,062.70
35	107,028.00	106,618.50	107,867.00
36	105,935.00	99,628.20	105,250.20
37	103,561.60	96,317.60	104,146.40
38	96,376.70	84,351.80	91,598.50
39	103,079.70	100,919.80	106,273.50
40	99,812.50	100,553.10	102,577.60
41	94,379.90	97,197.50	103,527.40
42	104,395.20	94,446.50	101,632.10
43	89,315.70	88,590.40	100,211.20
44	106,497.70	103,027.10	105,738.00
45	101,060.30	101,299.50	104,468.50
46	101,685.00	101,229.80	103,700.10
47	108,695.70	104,365.50	105,520.10
48	102,179.00	102,264.30	106,299.80
49	97,581.90	98,457.00	102,055.20
50	102,016.00	102,239.80	82,936.80
51	108,417.50	99,397.10	84,373.20
52	96,503.90	97,276.90	81,393.70
TOTAL	5,439,681	5,431,298	5,511,366

To test the hypothesis that the EOM weeks have higher volume, three following methods are applied; (1) comparing two variances, (2) comparing two means and (3) logit regression analysis. Before describing the details, we define the EOM period and the EOM weeks. Clearly, there is no formal definition so we consider the following two possibilities for the EOM period:

1. Period from the 26th until the 5th of the next month (EOM1)

2. Period from the 28th until the 3rd of the next month (EOM2).

We define EOM weeks to be the weeks which cover more than 50% of an EOM period. Then, each EOM week should cover six or more days of EOM1 and four or more days of EOM2. It turns out that EOM1 and EOM2 both produce the same set of EOM weeks for the desired time period from October 1, 1998 to September 30, 2001. There are 30 EOM weeks and 114 regular weeks among the 144 weeks under consideration.

Comparing two variances. To test whether the population variances of the distributions of EOM weeks and regular weeks are equal, an F-test is performed with the following null and alternative hypotheses

$$H_0 : \sigma_E^2 = \sigma_R^2, \qquad H_a : \sigma_E^2 \neq \sigma_R^2,$$

where σ_E^2 and σ_R^2 denote the variances for the distributions of EOM weeks and regulars weeks, respectively. The result, given in Table 2.4, shows that the p-value is too high to reject the null hypothesis of equal variances with typical type I error levels of 0.1 or 0.05.

Table 2.4: F-test result for comparing variances

Measure	Regular	EOM	Test result	
Sample mean	105,057	105,895	F-statistic	1.1450
Standard deviation	6,427	6,006	$F_{110,\,29,\,0.05}$	1.7033
Observations	111	30	$F_{110,\,29,\,0.025}$	1.8930
degree of freedom	110	29	p-value	0.3472

Comparing two means. Consider the following null and alternative hypotheses

$$H_0 : \mu_E = \mu_R, \qquad H_a : \mu_E > \mu_R,$$

where μ_E and μ_R denote the mean demand volume for EOM weeks and the mean for regular weeks, respectively. In this analysis, we performed a one-sided t-test for comparing two means, assuming equal variances for regular and EOM weekly demand based on the previous test result. The result is given in Table 2.5. The p-value is too high to reject the null hypothesis of equal means, with error levels of 0.1 or 0.05.

Table 2.5: t-test results for comparing means

Test results	
degree of freedom	139
t-statistics	0.6425
$t_{0.1}$	1.2877
$t_{0.05}$	1.6559
p-value	0.2608

Logit regression analysis. Logistic, or logit, regression is a well-known approach for evaluating the effects of independent variables on a qualitative dependent variable, such as the presence or absence of certain factors. Using an indicator EOM variable as the qualitative dependent variable (E) and the demand variable as the qualitative independent variable (D), we can use logit regression in an attempt to infer whether the EOM variable effects the demand.

Consider the following *logistic* function

$$E(D) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 D)}}. \tag{2.21}$$

The logit transformation is the natural logarithm of the so-called odd ratio:

$$\text{logit } E(D) = \ln\left(\frac{E(D)}{1 - E(D)}\right), \tag{2.22}$$

which yields

$$\text{logit } E(D) = \beta_0 + \beta_1 D. \tag{2.23}$$

With this model, we want to test the hypothesis that there is no effect of variable E ($H_0 : \beta_1 = 0$), or, in our problem, EOM weeks do not have larger volume than the other weeks. If H_0 is true, then the model given by Equation (2.23) reduces to

$$\text{logit } E(D) = \beta_0 \tag{2.24}$$

The computations are performed by SAS software [60] and the results are shown in Table 2.6, from which we can conclude that we cannot reject the null hypothesis that there is no "end of month effect" since the p-values are too high to reject. Thresholds for the p-value are 0.10 and 0.05 with the significance levels 90% and 95%, respectively.

Table 2.6: Logit regression analysis results

Test	χ^2	df	p-value
Likelihood ratio	0.2569	1	0.6122
Wald	0.2531	1	0.6149

The result of the above analysis is consistent with respect to indicate there is insufficient statistical evidence to suggest the presence of an EOM effect. Hence, we ill not separate data by the EOM and will use to estimate a single distribution govern to weekly demand volume.

2.4.2 Estimation of Weekly Demand

The demand in weeks 50-52 in 2001 is considerably lower than that of other weeks. The corresponding month is September 2001 when there was a terrible tragedy in New York City. In our analyses, these weeks are removed as outliers.

After the Christmas holiday season and the September 2001 outlier weeks are removed, the histogram of the historical data is drawn and the distribution of the weekly demand is estimated. The histogram and the probability density function of the estimated distribution are shown in Figure 2.2. The histogram reflects a normal distribution with a mean of 105,000 and a standard deviation of 6,310. The estimation is performed with a simulation software ARENA. The test results are summarized in Table 2.7.

Further test assesses the appropriateness of assuming that demand is

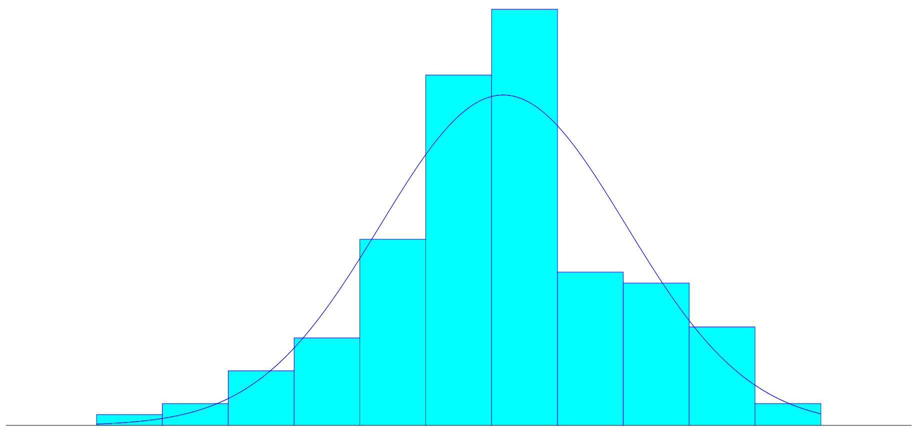


Figure 2.2: Histogram of weekly demand from 1999 to 2001 after removing outliers

Table 2.7: The result of estimation

Estimated distribution
Name: Normal(105000, 6310) Square error: 0.010288

Verification

χ^2 Test	Kolmogorov-Smirnov Test
Test statistic = 7.55 Corresponding p-value = 0.0584 Number of intervals = 6 Degrees of freedom = 3	Test statistic = 0.0674 Corresponding p-value $\geq$ 0.15

normally distributed, with the above parameters. We used a so-called Q-Q plot in which the observed values are plotted versus the quantiles of the assumed normal distribution (see, e.g., Johnson and Wichern [39]). The straightness of the Q-Q plot is measured by the correlation coefficient of the points in the plots defined by

$$r_Q = \frac{\sum_{j=1}^{n}(x_{(j)} - \bar{x})(q_{(j)} - \bar{q})}{\sqrt{(x_{(j)} - \bar{x})^2}\sqrt{(q_{(j)} - \bar{q})^2}}. \tag{2.25}$$

The quantiles $q_{(j)}$ are defined by the relation

$$P[N(\mu, \sigma) \leq q_{(j)}] = p_{(j)} = \frac{j - \frac{1}{2}}{n}$$

where $N(\mu, \sigma)$ denotes a normal with mean μ and variance σ. A Q-Q plot is the plot of pairs $(q_{(j)}, x_{(j)}), j = 1, ..., n$, where $x_{(j)}$ are the ordered observations. The Q-Q plot for the weekly demand is plotted in Figure 2.3.

The r_Q for the above Q-Q plot is calculated as 0.9924 with 141 observations, which is greater than the threshold 0.9906 with $\alpha = 0.05$ and 0.9922 with $\alpha = 0.10$. (These thresholds are interpolated.) Therefore, we cannot reject the normality with significance level $\alpha = 0.10$.

2.4.3 Equipment Scheduling for Daily Demand

The demand for a week can be realized from the distribution estimated in the previous section. Our model (see Section 2.3.2) takes as input a demand distribution which has a resolution in period over the course of one week, i.e., the distribution of $\widetilde{D}_{td}$, $t = 1, ..., n^T$, $d = 1, ..., 7$.

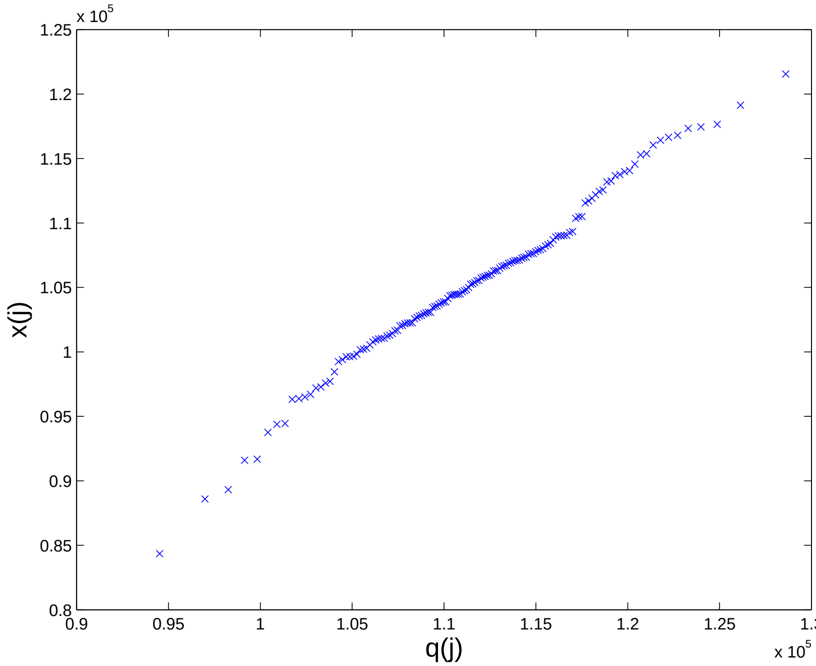


Figure 2.3: Q-Q plot for the weekly demand

To obtain the realizations of such distribution, we need an algorithm which takes mail volume for a week as an input and produces the number of workers per each period needed to satisfy the mail volume over a week. For this we use an equipment scheduling algorithm developed by Zhang [75]

The equipment scheduling algorithm developed by Zhang is to produce the required number of workers of each type to process the given volume of the mail while minimizing the labor costs associated with equipments. In this algorithm, a large-scale mixed-integer program is optimized sequentially using a three-phase methodology:

1. Solve the basic multicommodity network flow model to minimize the ending inventory.

2. Relax the volume of mail not processed in Step 1 for each operation and minimize number of full-time shifts.

3. Relax the number of shifts found in Step 2 and minimize a combination of the number of startups and the weighted sum of working periods.

In each phase, a separate criterion is optimized and the corresponding objective function value is used as a constraint in the next phase. Approximations, pre- and post-processing's are used to enhance the performance. As a result of this algorithm, the number of each shift for each machine is produced.

We give an equipment scheduling algorithm described above one observation of weekly demand estimated in Section 2.4.2 and obtain realizations of $\widetilde{D}_{td}$, $t = 1, ..., n^T$, $d = 1, ..., 7$, by adding up the numbers of workers over machines in each period from the result. So, we feed as many observations of our weekly distribution into this one-to-many mapping as we need in order to estimate the distribution of $\widetilde{D}_{td}$, which we will subsequently use.

Figure 2.4 shows an example of the daily demand generated by this algorithm. In this detailed demand, each day is divided into $n^T = 48$ periods since USPS uses 30 minute periods. For accounting purposes, the first period starts at midnight.

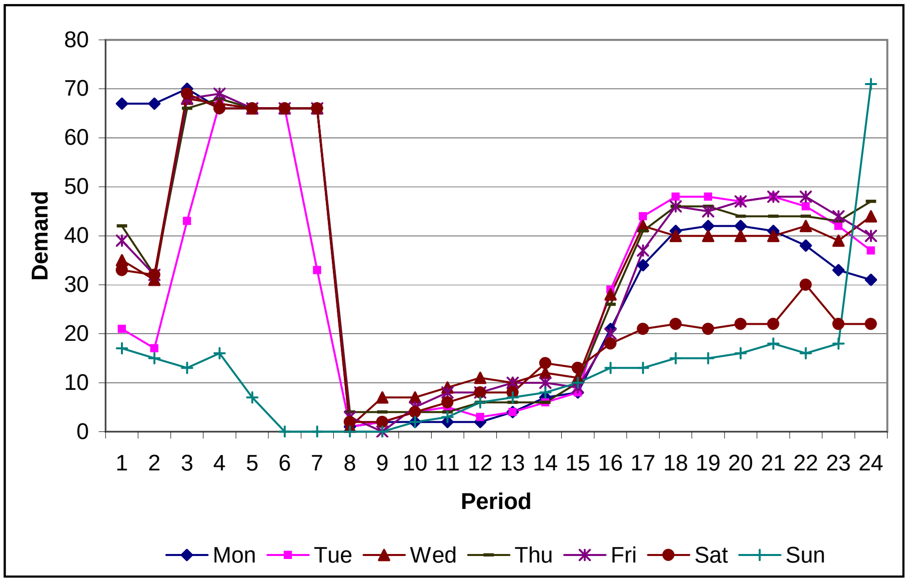


Figure 2.4: An example of demand per period over a day

2.5 Computational Results of SPSP

A series of experiments was performed to gain insight into the proposed models. We seek to understand their strengths and weaknesses, the value of the stochastic solution, and the accompanying computational effort. We begin with the SIP (2.19) in which all of the demand must be processed in the current period. Any demand that cannot be met is accounted for by additional FTRs via variables w_a^U and x_{fd}^U.

2.5.1 Data

In the analysis, the number of periods in a day is $n^T = 24$ and breaks are allocated to all shifts longer than 6 hours during the 4th or 5th period. The specific assignments are actually made by a post-processor after the values of the decision variables are found. The length of a full-time shift is 9 periods including the break and may start at any hour, implying $n^F = 24$. Similarly, part-time shifts may start at any hour, but their lengths may vary. Table 2.8 identifies the two cases examined. In the first case (S2), shifts of 4 and 5 periods are used so $n^P = 48$. In the second case (S4), shifts of length 4, 5, 6 and 7 periods are considered, giving $n^P = 96$ possibilities. Part-time shifts of length 7 have a break so, in reality, they represent 6 hours of work; however, they are somewhat more flexible than a 6-hour shift because of the break window.

The hourly wage rate for full-time workers was set at the national average of \$28.10 and \$25.85, respectively. This implies a type 1 over time rate of \$42.15/hr and a type 2 rate of \$56.20/hr. For casuals, we used an hourly rate of \$15.00. As described in Section 2.3.2, we measure the amount of unmet demand in units of full-time workers; their hourly rate was set to \$61.82, which is 10% above the highest rate—the type 2 overtime rate.

Table 2.8: Two sets of data

Set	Type	Start time	Shift length (periods)	Number of shifts
S2	Full-time	every hour	9*	24
	Part-time	every hour	4, 5	48
S4	Full-time	every hour	9*	24
	Part-time	every hour	4, 5, 6, 7*	96

* shifts that have a break

To complete the specification of parameter values, we set the lower bound on the ratio of full-time to part-time workers (ρ) to 8, overtime hours were limited to 6.25% of the regular hours associated with the permanent workforce ($\rho^{OT} = 0.0625$), and no more than 5.9% of the combined regular hours for FTRs and PTFs were allowed for casuals ($\rho^{C} = 0.059$). To use casual workers, at least 95% of the maximum overtime hours must be used ($\underline{\rho}^{OT} = 0.0594$). On average, the maximum amount of working hours for each PTF ($\bar{\theta}$) was set at 25 while the minimum number of hours required before casuals could be used ($\underline{\theta}$) was a function of the specific scenario, ranging from 14 to 20.

We approximate the distribution of $\widetilde{D}$ with a three-point distribution with realizations (scenarios) D^L, D^M and D^H and respective probability weights p^L, p^M and p^H. We use $D^L = D^M - 1.5\sigma$ and $D^H = D^M + 1.5\sigma$, where σ is the standard deviation of weekly demand. The masses $p^L = p^H = 0.22$ and $p^M = 0.56$ were chosen so that the three-point approximating distribution matched the variance of the three-year data set. The value of D^M was selected as that of the 'best week' scenario from Bard (2004). The Figure 2.4 shows $F(D^M)$ and Table 2.9 describes $F(D^L)$, $F(D^M)$ and $F(D^H)$, which, along with their respective weights, represented the three-scenario distribution. As indicated in that table, higher mail volumes don't always translate into higher labor requirements per period, primarily due to the nonlinear nature of F, i.e., Zhang's scheduling algorithm.

All models were implemented in Java and solved with the CPLEX 9.0

Table 2.9: Input demand data (workers per period)

(a) Demand scenario H

Period	1	2	3	4	5	6	7	8	9	10	11	12
Mon	68	68	69	67	66	66	33	0	1	4	2	3
Tue	29	28	64	35	35	66	33	2	6	2	2	2
Wed	35	49	69	66	66	66	33	7	10	9	10	10
Thu	38	51	69	66	66	66	33	2	5	7	5	12
Fri	34	51	70	66	66	66	33	1	4	7	9	12
Sat	56	68	69	62	66	66	33	3	6	7	4	7
Sun	17	15	17	15	8	0	0	0	3	4	7	6
Period	13	14	15	16	17	18	19	20	21	22	23	24
Mon	5	10	12	17	33	45	42	44	44	39	33	37
Tue	2	8	8	19	44	51	52	56	56	52	52	47
Wed	8	10	10	14	35	46	51	52	54	52	50	45
Thu	10	14	16	20	44	50	52	50	49	43	40	40
Fri	12	8	8	12	32	45	56	55	55	58	57	48
Sat	6	9	13	20	22	23	25	29	29	30	31	26
Sun	7	6	5	6	17	18	21	24	26	45	66	71

(b) Demand scenario M

Period	1	2	3	4	5	6	7	8	9	10	11	12
Mon	67	67	70	66	66	66	66	1	2	2	2	2
Tue	21	17	43	67	66	66	33	1	2	4	5	3
Wed	35	31	68	67	66	66	66	1	7	7	9	11
Thu	42	32	66	68	66	66	66	4	4	4	4	6
Fri	39	32	68	69	66	66	66	3	0	5	8	8
Sat	33	32	69	66	66	66	66	2	2	4	6	8
Sun	17	15	13	16	7	0	0	0	0	2	3	6
Period	13	14	15	16	17	18	19	20	21	22	23	24
Mon	4	7	8	21	34	41	42	42	41	38	33	31
Tue	4	6	8	29	44	48	48	47	48	46	42	37
Wed	10	12	11	28	42	40	40	40	40	42	39	44
Thu	6	6	10	26	41	46	46	44	44	44	43	47
Fri	10	10	9	20	37	46	45	47	48	48	44	40
Sat	8	14	13	18	21	22	21	22	22	30	22	22
Sun	7	8	10	13	13	15	15	16	18	16	18	71

Table 2.9: Input demand data (continued)

(c) Demand scenario L

Period	1	2	3	4	5	6	7	8	9	10	11	12
Mon	68	68	69	67	66	66	33	1	2	2	4	3
Tue	24	21	44	68	35	66	33	0	0	1	0	2
Wed	36	42	57	67	39	66	33	3	5	6	6	6
Thu	31	37	61	66	39	66	33	1	5	4	3	7
Fri	31	50	69	67	66	66	33	0	2	2	2	5
Sat	28	50	70	66	66	66	33	2	2	2	3	7
Sun	18	14	16	14	7	0	0	0	1	1	5	5
Period	13	14	15	16	17	18	19	20	21	22	23	24
Mon	4	4	5	6	21	37	38	36	32	38	35	30
Tue	4	8	8	17	34	43	43	44	43	41	38	37
Wed	5	8	10	17	35	36	39	42	42	38	37	36
Thu	7	9	9	17	35	36	40	38	37	38	40	38
Fri	5	8	8	20	40	43	44	44	44	42	41	41
Sat	7	8	11	13	18	19	18	22	22	32	21	20
Sun	5	3	2	8	8	13	18	18	19	22	41	70

callable libraries in the form of the large-scale equivalent MIP described in Section 2.3.1. A 0.5% optimality gap was set for the termination criterion. The computations were performed on a Dell Precision 530 workstation with dual 1.8 GHz Pentium Xeon CPUs and 1GB memory running SuSE Linux, although only one processor was used.

2.5.2 Results for Basic Model

The computational results in this section are based on a single week of operations. As explained in Section 2.3.2, to determine the total labor cost for the year, weekly costs can simply be multiplied by 52. The number of FTRs reported along with their schedules are invariant, except for overtime. The weekly schedules of the PTFs and casuals depend on the demand realization.

The solutions obtained for model (2.19)-(2.20) for cases S2 and S4 are summarized in Table 2.10. The x_f and w_a columns identify the total number of full-time shifts and the total number of FTRs, respectively. The subscripts f and a are only included as a reference to the original names of the variables. Similarly, v_b identifies the total number of part-timers and y_p the total number of PTF shifts. The number of type 1 and type 2 overtime assignments are given, respectively, by μ_f^1 and μ_f^2, the total number of casual shifts is γ_p, and the total number of FTRs needed to cover unmet demand is listed under w_a^U. Recall that the model is designed to ensure that PTF hours and overtime are used before casuals.

Regarding the various output measures, the *EV* solution is obtained by solving the deterministic problem with the random demand $[\widetilde{D}_{td}]$ replaced by its mean value $[\bar{D}]$. The RP solution considers all three scenarios simultaneously using the demand data in Table 2.9. WS(H) refers to the deterministic problem (i.e., one-scenario problem) under scenario H, WS(M) to the deterministic problem under scenario M, and WS(L) to the deterministic problem under scenario L. The WS value is the mean of the objective function values of these three wait-and-see values. The numbers in parentheses following the results for VSS and $EVPI$ indicate the fraction by which VSS and $EVPI$ exceed the objective function value of the recourse problem RP. Note that the $WS(L)$ value obtained for the S4 shift set was marginally greater than the value obtained for the S2 shift set even though the S4 model has a larger second-stage feasible region. This anomaly is explained by the fact that the 0.003% difference in objective function values is well within the 0.5% optimality gap used as the stopping criterion.

The number of full-time and part-time workers associated with the RP solution was larger than the number obtained by solving the EV problem. EEV, which uses the first stage solution of EV, produced unmet demand for both data sets. For S2, VSS was \$5,507 or 5% of the RP value, indicating that expected labor costs can be reduced by \$5,507 per week by solving RP rather than EV and adjusting the composition of the workforce accordingly. For S4, the cost savings were slightly greater, amounting to \$6,519 per week or 4% of the RP value. In both cases, the RP solution called for a larger permanent workforce than the EV solution, but the need for more costly overtime and

Table 2.10: Results for the various models

Shift set	Problem	Objective value	1st stage var			2nd stage var					
			w_a	x_f	v_b	D^{ω}	μ_f^1	μ_f^2	y_p	γ_p	w_a^U
S2	EV	151,498	111	555	13	$\bar{D}$	150	9	65	78	0
	EEV	165,694	111	555	13	D^H	142	18	65	75	10
						D^M	159	2	65	76	5
						D^L	149	6	51	76	0
	RP	157,187	116	580	14	D^H	144	29	70	78	0
						D^M	162	4	70	76	0
						D^L	136	25	50	38	0
	WS(H)	156,411	114	570	14	D^H	140	23	70	80	0
	WS(M)	153,115	112	560	14	D^M	159	7	70	79	0
	WS(L)	140,788	103	515	12	D^L	107	39	59	72	0
	WS	151,128									
	VSS	5,507	(0.05)								
	EVPI	6,059	(0.04)								
S4	EV	151,460	111	555	13	$\bar{D}$	149	10	65	78	0
	EEV	163,479	111	555	13	D^H	140	23	64	73	11
						D^M	161	6	65	79	4
						D^L	139	16	48	76	0
	RP	156,960	116	580	14	D^H	154	20	70	80	0
						D^M	162	3	69	76	0
						D^L	139	23	50	43	0
	WS(H)	156,404	114	570	14	D^H	138	26	69	79	0
	WS(M)	153,048	112	560	14	D^M	160	0	70	72	0
	WS(L)	140,792	103	515	12	D^L	108	39	60	80	0
	WS	151,090									
	VSS	6,519	(0.04)								
	EVPI	5,870	(0.04)								

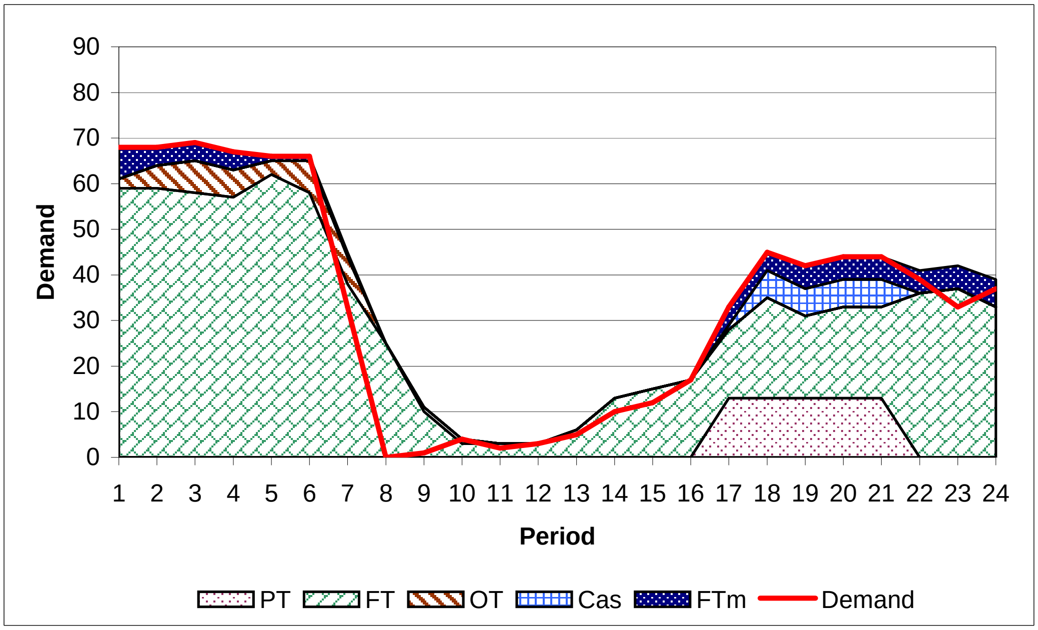
the use of ‘extra’ workers to cover the unmet demand was reduced.

The expected value of perfect information ($EVPI$) is the second measure of interest. From the results, it is evident that we should be willing to pay up to \$6,059 to gather information about the future for S2 and up to \$5,870 with S4.

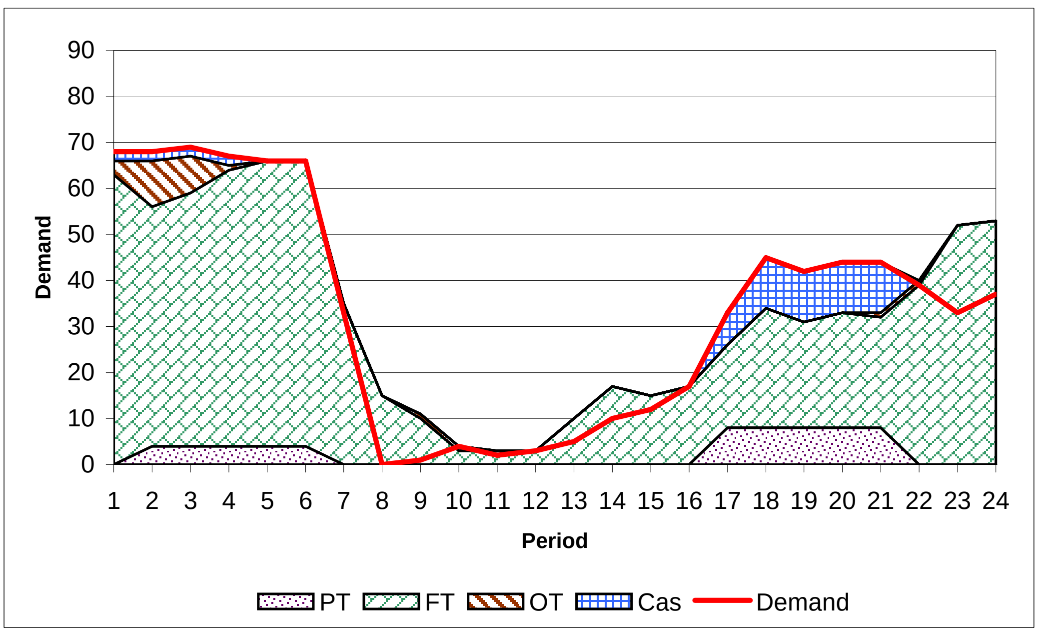
Figure 2.5 depicts the period-by-period details of the EEV and RP solutions for Monday under the high demand scenario for S2. PT refers to the number of part-time shifts, FT to the number of full-time shifts, Cas to the number of casual workers, and OT to the total number of type 1 and 2 overtime shifts. In the case of EEV, FTm denotes the number of workers assigned to full-time shifts to cover unmet demand. The solid line indicates the demand over 24 hours, while the other lines indicate the aggregate number of shifts by category. The first category corresponds to the part-timers and the second to the full-timers. The next two areas in the figure correspond to overtime and casual workers, respectively.

The darkly shaded area at the top of Figure 2.5a corresponds to the additional full-time workers hired to cover unmet demand. As can be seen, the EEV solution has a manpower shortage during periods 1 to 6 and 16 to 22. For the problems that we considered, no such additional workers were part of the RP or WS solutions.

Table 2.11 gives the number of variables and constraints associated with each MIP as well as the amount of (clock) time used in the computations.



(a) EEV solution under high demand scenario



(b) RP solution under high demand scenario

Figure 2.5: EEV and RP solutions for Monday for S2 data set

As expected, RP took considerably longer to solve than the other problems because it is the only model that fully embodies the three scenarios. Although EEV also addresses the three scenarios simultaneously, the fact that the first stage variables are fixed in this problem allows the second stage to be solved separately for each scenario.

Table 2.11: Problem dimensions and computation times

Shift set	Problem	Problem size†		CPU time (sec)
		Variables	Constraints	
S2	EV	1542	1105	101
	EEV	2922	1858	5
	WS(H)	1542	1105	95
	WS(M)	1542	1105	71
	WS(L)	1542	1105	401
	RP	4258	3121	1661
S4	EV	2214	1160	311
	EEV	4686	2190	3
	WS(H)	2214	1160	193
	WS(M)	2214	1160	26
	WS(L)	2214	1160	330
	RP	6274	3286	3411

† after CPLEX preprocessing

Figure 2.6 depicts the RP objective function values for a range of full-time to part-time ratios (i.e., parameter ρ). The results indicate that as the ratio increases the objective function value increases at a decreasing rate. Lower ratios mean more flexibility and hence lower costs, and the lower the ratio, the greater the marginal benefit.

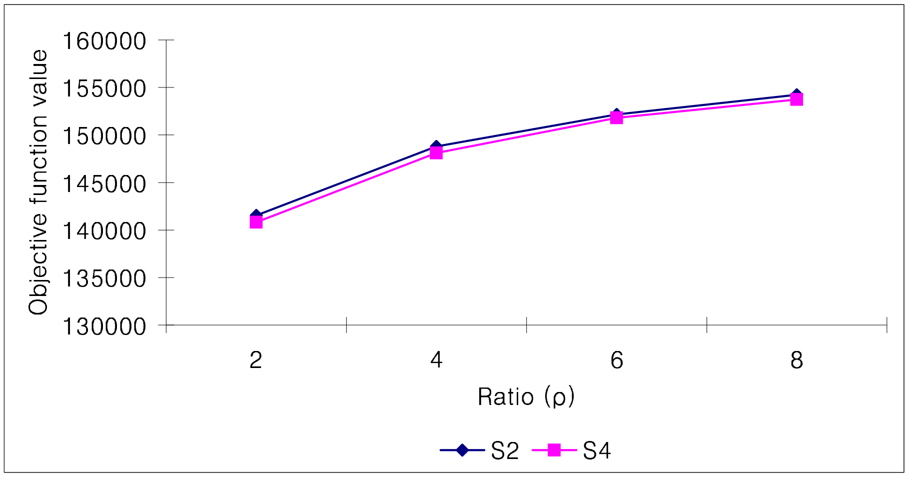


Figure 2.6: Parametric results for RP as a function of the full-time to part-time ratio

2.5.3 Heuristic Approach to the Recourse Problem

The deterministic equivalent form of (2.19)-(2.20) becomes dramatically more difficult to solve as more scenarios are included in the analysis. To ease the computational burden, a *target heuristic* is developed aimed at finding good feasible solutions. The idea is to first solve the LP relaxation of (2.19)-(2.20) to obtain a target solution and then find a "nearby" integer point. This is achieved by solving an IP whose objective function consists of the following two terms: (i) the sum of the absolute deviations from the LP solution, and (ii) the original objective function scaled by a constant $\varepsilon \geq 0$. When ε is small, staying close to the LP solution is primary and minimizing the original objective function is secondary. When ε is large these roles are reversed.

To implement this approach, let X be an appropriately dimensioned vector denoting a solution to the problem (2.19)-(2.20), i.e., $X = (w_a, v_b, x_{fd}, y_{pd}, \beta_{td}, \mu^1_{fd}, \mu^2_{fd}, \gamma_{pd}, w^U_a, x^U_{fd})$. Also, let $f(X)$ be the original objective function (2.19a) and X^{LP} a solution to the LP relaxation. Then a heuristic solution can be obtained by solving the following IP

$$\begin{aligned} z = \quad & \min_X \quad |X - X^{LP}| + \varepsilon f(X) \\ & \text{s.t.} \quad (2.19b) - (2.20r), \end{aligned} \tag{2.26}$$

where X^{LP} is given (of course, it is necessary to linearize the first term in the objective function of (2.26) before solving). When the above problem is solved, the CPLEX parameter was set to stop after finding the 5th integer feasible solution.

Table 2.12 lists the objective function values $f(X)$ and computation times for various values of the parameter ε. As expected, the costs were higher than those in Table 2.10 but the solution times were only a small fraction of those associated with the full problem. When ε was small, the target heuristic produced solutions having costs about 6% higher than the optimum while achieving time reductions ranging from 80 and 92%. When ε was large the expected cost of the solution was much closer to that of RP, and surprisingly we still achieved considerable reduction in computational effort.

Table 2.12: Results for the target heuristic

Data set	ε	Objective cost	Cost increase (%)	Time (sec)	Time reduction (%)
S2	0	166,762	6.2	400	87.4
	10^{-8}	166,892	6.3	324	89.8
	10^{-7}	166,892	6.3	324	89.8
	10^{-6}	164,929	5.0	254	92.0
	10^{5}	159,681	1.6	375	88.2
	10^{6}	158,763	1.1	316	90.1
	10^{7}	158,614	0.9	444	86.1
S4	0	172,875	10.1	552	83.8
	10^{-8}	167,308	6.6	680	80.1
	10^{-7}	167,308	6.6	680	80.1
	10^{-6}	166,920	5.7	436	87.2
	10^{5}	157,681	0.6	363	86.7
	10^{6}	157,763	0.6	364	89.3
	10^{7}	160,050	2.0	368	89.2

Chapter 3

Column Generation

Stochastic programming enables the model to capture more realistic factors. However, it becomes impractical when the number of scenarios grows and it gets more complicated when the model has integrality constraints. Aiming to maneuver this, decomposition techniques have been applied to solve a stochastic programming problem. In this chapter, we review column generation and implement the column generation version of the models described in the previous chapter.

3.1 Column Generation Review

3.1.1 Column Generation Formulation

Consider the following LP.

$$
\begin{aligned}
z^* = \min_{x_1, x_2, ..., x_l} \quad & \sum_{i=1}^{l} c_i x_i \\
\text{s.t.} \quad & A_i x_i = b_i, \quad i = 1, ..., l && \text{(3.1a)} \\
& \sum_{i=1}^{l} D_i x_i = d && \text{(3.1b)} \\
& x_i \geq 0, \quad i = 1, ..., l.
\end{aligned}
$$

Let $S_i = \{x_i : A_i x_i = b_i, x_i \geq 0\}$, $i = 1, ..., l$, and assume S_i is not empty and bounded. Let $x_i^k, k \in K_i$, index all extreme points of S_i. Then the problem (3.1) can be rewritten as follows.

$$\begin{aligned}
\min_{\lambda} \quad & \sum_{i=1}^{l} c_i \left(\sum_{k \in K_i} x_i^k \lambda_i^k \right) \\
\text{s.t.} \quad & \sum_{i=1}^{l} D_i \left(\sum_{k \in K_i} x_i^k \lambda_i^k \right) = d \\
& \sum_{k \in K_i} \lambda_i^k = 1, \quad i = 1, ..., l \\
& \lambda_i^k \geq 0, \quad k \in K_i, \, i = 1, ..., l.
\end{aligned}$$

The above problem is called the (full) *master problem*. Let λ_i^{k*} be the solution to the full master problem with all of the extreme points in $S_i, i = 1, ..., l$. Then $x_i^* = \sum_{k \in K_i} x_i^k \lambda_i^{k*}, i = 1, ..., l$, solves the problem (3.1). Of course, it is neither practical nor desirable to explicitly enumerate all extreme points of K_i, $i = 1, ..., l$. Instead we start with modest-sized subsets $K_i' \subset K_i, i = 1, ..., l$. This leads to the so-called restricted master problem (RMP).

$$\begin{aligned}
\bar{z} = \min_{\lambda} \quad & \sum_{i=1}^{l} c_i \left(\sum_{k \in K_i'} x_i^k \lambda_i^k \right) \\
\text{s.t.} \quad & \sum_{i=1}^{l} D_i \left(\sum_{k \in K_i'} x_i^k \lambda_i^k \right) = d \qquad (3.2a) \\
& \sum_{k \in K_i'} \lambda_i^k = 1, \quad i = 1, ..., l \qquad (3.2b) \\
& \lambda_i^k \geq 0, \quad k \in K_i', \, i = 1, ..., l.
\end{aligned}$$

The above RMP may not be feasible in an early stage, e.g., obviously (3.2) is infeasible when no columns is present. If there exist columns at hand that make the problem (3.2) feasible, we use them. Otherwise general methods for a simplex method such as M-method or two-phase method can be applied.

The columns are generated using the pricing problems which is based on the reduced cost for λ. The reduced cost for λ_i is

$$\begin{aligned} \bar{c}_i &= c_i x_i - \pi D_i x_i - \alpha_i \\ &= (c_i - \pi D_i)\, x_i - \alpha_i. \end{aligned}$$

where π is the dual variable associated with (3.2a) and α_i is with (3.2b). So based on the above reduced costs, we construct the following pricing problem.

$$\begin{aligned} v_i = \min_{x_i} \quad & (c_i - \pi D_i) x_i \\ \text{s.t.} \quad & A_i x_i = b_i \\ & x_i \geq 0. \end{aligned} \tag{3.3}$$

If $v_i - \alpha_i \geq 0$, for all $i = 1, ..., l$, we are optimal. Otherwise, pick a solution $\hat{x}_i$ to (3.3) having $v_i - \alpha_i < 0$ and add a column $\hat{x}_i = (\hat{x}_{i1}, \hat{x}_{i2}, ..., \hat{x}_{in})^T$ to the restricted master problem.

The lower bound can be provided by using the pricing problem based on the Lagrangian relaxation.

Theorem 3.1. *(Bertsimas and Tsitsiklis [13]) Let $\bar{z}$ denote the objective function value of the restricted master problem. Also let α_i denote the dual solution associated with the convexity constraint and let v_i be the objective function value of the ith pricing problem. Finally let $\underline{z} \equiv \bar{z} + \sum_{i=1}^{l}(v_i - \alpha_i)$. Then*

$$\underline{z} \leq z^* \leq \bar{z}.$$

Now we can summarize the column generation procedure in Figure 3.1.

Procedure CG

Step 0 Initialize RMP (3.14) with a set of columns $\begin{pmatrix} c_i x_i^k \\ D_i x_i^k \end{pmatrix}$, $k \in K_i'$ for $i = 1, ..., l$.

Step 1 Solve the LP relaxation of the RMP (3.14) to obtain optimal value $\bar{z}$, primal solution λ, and dual solution (π, α_i) for $i = 1, ..., l$.

Step 2 For $i = 1, .., l$, solve the pricing problem (3.15) to obtain v_i, and $\hat{x}_i$.

Step 3 If $v_i \geq \alpha_i$ for all $i = 1, ..., l$, the stop and report $\bar{z}$ and x^* where $x_i^* = \sum_{k \in K'} x_i^k \lambda_i^k$, $i = 1, ..., l$.

Step 4 For $i = 1, ..., l$, if $v_i < \alpha_i$ then add an element to index set K_i' for new column $\begin{pmatrix} c_i \hat{x}_i \\ D_i \hat{x}_i \end{pmatrix}$.

Goto step 1.

End procedure

Figure 3.1: A column generation procedure.

3.1.2 Stabilization

It is well known that a CG approach converges slow. Gilmore and Gomory found that the algorithm slowed down toward the end when they tested on the cutting stock problem [30]. It can be observed that the dual values oscillate, which produces poor quality columns and poor convergence. Many have been trying to "stabilize" dual values by preventing them from going "too far" from iteration to iteration, which includes defining boxes for bounds on dual values, using a weighted sum of current dual values and previous ones and taking 'interior' dual values rather than extreme dual solutions. In this section we will review an interior point method by Rousseau, Gendreau, and Feillet [59] or Goffin and Vial [31] and a hybrid method proposed by du Merle et al. [25].

Interior Point method Let $\hat{\lambda}_i, i = 1, ..., l$, be the solution of (3.2) at some iteration and let $K_i'^*$ denote $K_i' \cap \{k : \hat{\lambda}_i^k > 0\}$. Now consider the dual of (3.2).

$$\begin{aligned} \max_{\pi,\alpha} \quad & \pi d + \sum_{i=1}^{l} \alpha_i \\ \text{s.t.} \quad & \pi D_i x_i^k + \alpha_i \leq c_i x_i^k, \quad k \in K_i', i = 1, ..., l. \end{aligned} \tag{3.4}$$

By the complementary slackness conditions, the solution to the following problem is also an optimal solution to (3.4) with different objective function value where u and u_i's are uniform random variables between 0 and 1. Note that u is a properly dimensioned vector associated with π.

$$\begin{aligned} \max_{\pi,\alpha} \quad & u\pi + \sum_{i=1}^{l} u_i \alpha_i \\ \text{s.t.} \quad & \pi D_i x_i^k + \alpha_i = c_i x_i^k, \quad k \in K_i'^*, i = 1, ..., l, \\ & \pi D_i x_i^k + \alpha_i \leq c_i x_i^k, \quad k \in K_i' \backslash K_i'^*, \; i = 1, ..., l. \end{aligned} \tag{3.5}$$

If the problem (3.13c) is solved with a simplex based method then its optimal solution will constitute an extreme point of (3.4). Different instances of u and u_i's can generate multiple optimal solutions to (3.4). If (3.4) has a unique optimal solution, then the problem (3.13c) will produce the unique optimal solution even with different instances of u and u_i's. Note that $1-u$ is also uniformly distributed between 0 and 1. Hence, various instances of u and u_i's are generated and the above problem is solved with pairs of (u, u_i) and $(1-u, 1-u_i)$. By doing this way, interior dual values can be produced by taking average of all obtained extreme points.

Hybrid Method The hybrid method combines the ideas of perturbations of equality constraints and limiting the bounds of dual variables. Perturbation avoids the dual degeneracy and dual bounds keep from instability in the dual solutions that are jumping from one extreme value to another.

To implement this method, the following problem is used as the restricted master problem, instead of the problem (3.2).

$$\begin{aligned}
\min_{\lambda, y^+, y^-} \quad & \sum_{i=1}^{l} c_i \left(\sum_{k \in K_i'} x_i^k \lambda_i^k \right) + \kappa^+ y^+ - \kappa^- y^- \\
\text{s.t.} \quad & \sum_{i=1}^{l} D_i \left(\sum_{k \in K_i'} x_i^k \lambda_i^k \right) + y^+ - y^- = d && \text{(3.6a)} \\
& \sum_{k \in K_i'} \lambda_i^k = 1, \quad i = 1, ..., l && \text{(3.6b)} \\
& y^+ \leq \epsilon^+ && \text{(3.6c)} \\
& y^- \leq \epsilon^- && \text{(3.6d)} \\
& \lambda_i^k, y^+, y^- \geq 0, \quad k \in K_i', \, i = 1, ..., l.
\end{aligned}$$

This model modifies constraint (3.2a), allowing infeasibilities that are limited by ϵ_j^+ and ϵ_j^- for each job $j \in J$. Such violations are penalized in the objective function using κ_j^+ and κ_j^-, $j \in J$. The dual of (3.6) provides insight as

$$
\begin{aligned}
\max_{\pi,\alpha,\delta^+,\delta^-} \quad & \pi d + \sum_{i=1}^{l} \alpha_i - \left(\epsilon^+\delta^+ + \epsilon^-\delta^-\right) \\
\text{s.t.} \quad & \pi D_i x_i^k + \alpha_i \le c_i x_i^k, \quad k \in K_i',\ i = 1, ..., l \\
& \pi - \delta^+ \le \kappa^+ \qquad (3.7a) \\
& -\pi - \delta^- \le -\kappa^- \qquad (3.7b) \\
& \delta^+, \delta^- \ge 0,
\end{aligned}
$$

where π, α, δ^+ and δ^- are the dual variables associated with (3.6a)-(3.6d), respectively. Constraints (3.7a) and (3.7b) imply

$$\kappa^- - \delta^- \le \pi \le \kappa^+ + \delta^+.$$

Thus choosing π outside the (trust) region $[\kappa^-, \kappa^+]$ is possible with δ^+ and δ^- but such deviations are penalized in (3.7)'s objective function using ϵ^+ and ϵ^-. These parameters are updated dynamically. A natural choice for κ^+ and κ^- is to use the dual solution from the previous iteration, i.e., $\kappa^+ = \kappa^- = \pi$. If the dual variables, π, handed to the pricing problems failed to produce new columns then we decrease ϵ^+ and ϵ^- and otherwise we increase these dual penalty coefficients.

Due to the perturbation of constraints in (3.6a), the lower bound, described in Section 3.1.1, is not valid. When $\epsilon^+ = \epsilon^- = 0$ and model (3.6) reduces to the RMP (3.2) we can, of course, use the lower bound. When applying stabilization we replace Step 3 in the column generation procedure of Figure 3.1 with the following step, where $r < 1$ is a predefined constant.

Step 3 If $v_i \geq \alpha_i$ for all $i = 1, ..., l$ then.

If $\epsilon_j^+ = \epsilon_j^- = 0$, then stop and report z_{LP} and x^* where $x_i^* = \sum_{k \in K_i'} x_i^k \lambda_i^k$, for $i = 1, ..., l$.

Otherwise update $\epsilon^+ \leftarrow r\,\epsilon^+,\ \epsilon^- \leftarrow r\,\epsilon^-$ and $\kappa^+ \leftarrow \pi,\ \kappa^- \leftarrow \pi$ where π is the dual solution obtained at step 1. Goto step 1.

3.1.3 Column Generation for Mixed Integer Programming

Suppose the problem (3.1) contains one or more integer variables. Then the master problem can be presented as

$$\begin{aligned}
\bar{z} = \min_{\lambda_1, \lambda_2, ..., \lambda_l} \quad & \sum_{i=1}^{l} c_i \left(\sum_{k \in K_i'} x_i^k \lambda_i^k \right) \\
\text{s.t.} \quad & \sum_{i=1}^{l} D_i \left(\sum_{k \in K_i'} x_i^k \lambda_i^k \right) = d && (3.8a) \\
& \sum_{k \in K_i'} \lambda_i^k = 1, \quad i = 1, ..., l && (3.8b) \\
& \lambda_i^k \in \{0, 1\}, \quad k \in K_i', \ i = 1, ..., l,
\end{aligned}$$

where $x_i^k, k \in K_i'$, are the (integer) feasible points of $Z_+ \cap S_i = \{x_i : A_i x_i = b_i, x_i \geq 0\}$. The pricing problems are, for $i = 1, ..., l$,

$$\begin{aligned}
v_i = \min_{x_i} \quad & (c_i - \pi D_i) x_i && (3.9) \\
\text{s.t.} \quad & x_i \in S_i \cap Z_+
\end{aligned}$$

We solve the LP relaxation of the master problem (3.8) until we are not able to add a column by solving pricing problems (3.9). When no column is added, we apply branch-and-bound procedure. Together with column generation, this procedure is called a *branch-and-price* (B&P) procedure.

Suppose we solve the RMP as an IP with current columns. Similar to a standard branch-and-bound procedure, an algorithm repeatedly chooses a fractional variable, adds a branching constraint and solves the problem until the lower and upper bounds are met. However it is possible that a standard branch-and-bound procedure would not provide the optimal columns in a B&P approach. When a fractional variable is branched, it may be the case that there exists a feasible column with a negative reduced cost but such a column does not present in the current RMP. Therefore feasible columns should be generated after branching and it does not guarantee to find an optimal solution by solving the problem (3.8) using the standard branch-and-bound, i.e., by solving the 0-1 integer problem (3.8) by CPLEX. Also many studies show that, when branching, it is better to use a variable in the original problem rather than one in the RMP, i.e., x_{ij} is better to use than λ_i [38]. Note that x_{ij}'s are data in the RMP (3.14). Identical restrictions by the branching constraints have to be applied to the pricing problems too. Otherwise existing columns in the master problem could be generated again [8, 72].

The lower bound in Theorem 3.1 can be extended for a mixed integer program.

Proposition 3.1. *Let z denote the objective function value of the LP relaxation of the restricted master problem. Also let α_i denote the dual optimal solution associated with the convexity constraint and let v_i be the objective* *function value of the* i*th pricing problem. Finally let* $\underline{z} \equiv z + \sum_{i=1}^{l}(v_i - \alpha_i)$*. Then*

$$\underline{z} \leq z^*.$$

Proof. Let π and α_i be the optimal dual solutions to the restricted master problem. Consider the Lagrangian relaxation of the problem (3.1) dualizing the constraint (3.1b) with a constant π.

$$\begin{aligned} \hat{z} = \min_{x_1, x_2, \ldots, x_l} \quad & \sum_{i=1}^{l}(c_i - \pi D_i)x_i + \pi d \\ \text{s.t.} \quad & A_i x_i = b_i, \quad i = 1, \ldots, l \\ & x_i \geq 0, \quad i = 1, \ldots, l \end{aligned} \tag{3.10}$$

Any feasible solution to (3.1) is feasible, but not necessary optimal, to (3.10) and the objective function value is the same because the dualized constraints are equality constraints and hence it is a relaxation, i.e., $z^* \geq \hat{z}$. Note $\hat{z} = \sum_{i=1}^{l} v_i + \pi d$.

$$\begin{aligned} z^* &\geq \hat{z} \\ &= \pi d + \sum_{i=1}^{l} v_i \\ &= \left(\pi d + \sum_{i=1}^{l} \alpha_i\right) + \left(\sum_{i=1}^{l} v_i - \sum_{i=1}^{l} \alpha_i\right) \\ &= \bar{z} + \sum_{i=1}^{l}(v_i - \alpha_i) \end{aligned}$$

The last equality holds due to the strong duality of the restricted master problem. □

Branch-and-Price Algorithm The general branch-and-price algorithm is summarized as follows. Let P be the LP relaxation of (3.8) and S be its feasible region.

Step 0 (Solve the root node. Do the procedure **CG** in Figure 3.1.)

Solve P with S. If the solution is integer, then terminated the algorithm. Otherwise find a fractional value of $x_i = \sum_{k \in K_i'} x_i^k \lambda_i^k$ and add the problems $P \cap \{x_i \leq \lfloor x_i \rfloor\}$ and $P \cap \{x_i \geq \lceil x_i \rceil\}$ to the list. Set $\hat{x} = 0$.

(Repeat the followings steps while the list is not empty.)

Step 1 Choose P from the list. If the list is empty, then stop and report $\hat{x}$ as an optimal solution with the objective function value of $\bar{z}$.

Step 2 (Solve the restricted master problem)

- Solve P to obtain the objective function value z and the primal λ and dual (π, α) solutions.

- If P has an integer solution and $z \leq \bar{z}$, then set $\bar{z} = z$ and $\hat{x}_i = \sum_{k \in K_i'} x_i^k \lambda_i^k$, $i = 1, ..., l$. Fathom the node and goto step 1.

- If P is infeasible, fathom the node and goto step 1.

Step 3 (Solve the pricing problems)

- For all i, solve a pricing problem with π to obtain the objective value v_i. If $v_i - \alpha < 0$, add a solution x_i to the restricted master problem as a column.

- Set $\underline{z} = z + \sum_{i \in I}(v_i - \alpha_i)$. If $\underline{z} \geq \bar{z}$, fathom the node and goto Step 1.

- If no column is added for all i, then goto step 4 and, otherwise, goto step 2.

Step 4 (Fathoming or branching)

If $z \geq \bar{z}$, fathom the node by bound. Otherwise find a fractional solution $x_i = \sum_{k \in K_i'} x_i^{(k)} \lambda_i^k$ and add problems $P \cap \{x_i \leq \lfloor x_i \rfloor\}$ and $P \cap \{x_i \geq \lceil x_i \rceil\}$ to the list. Goto step 1.

For the root node at step 0, early termination criterion may be adopted with a tolerance ε, using the lower and upper bounds. We stop step 0 when the following is satisfied.

$$\bar{z} - \underline{z} \leq \min\{|\bar{z}|, |\underline{z}|\} \cdot \varepsilon$$

3.1.4 Computational Results for Stabilization

The performance of stabilization is tested with the stochastic facility location problem (SFLP). The model for the column generation of SFLP is adopted from Silva and Wood [63]. SFLP is to identify the best locations for capacitated production facilities that will produce and ship products to customers to meet their demands. In this model, demand and capacity are known in distributions. The purpose of this experiment is to see the performance of the stabilization methods and hence the branch-and-bound part of the algorithm is skipped, i.e., procedure **CG** described in Figure 3.2 is performed. The following notation is used to describe the problem.

Indices

$i \in I$ potential facility locations

$j \in J$ customers

$s \in S$ scenarios

Parameters

c_{ij} average total cost for supplying the whole demand of customer j from facility i

f_i fixed cost for installing a facility at location i

d_j^s demand of customer j under scenario s

u_i^s capacity of facility j under scenario s

ρ_i^s penalty for unmet demand for facility i under scenario s

p^s probability of scenario s

Decision variables

x_{ij} 1 if customer j is assigned to facility j; 0 otherwise

z_i 1 if facility i is installed; 0 otherwise

y_i^s the amount of unmet demand for facility i under scenario s

Then the model can be presented as

$$
\begin{aligned}
\min_{x,z,y} \quad & \sum_{i\in I} f_i z_i + \sum_{i\in I}\sum_{j\in J} c_{ij}x_{ij} + \sum_{s\in S}\sum_{i\in I} p^s \rho_i^s y_i^s \\
\text{s.t.} \quad & \sum_{i\in I} x_{ij} = 1, \quad \forall j \in J && \text{(3.11a)} \\
& -z_i + x_{ij} \leq 0, \quad \forall i \in I,\, j \in J && \text{(3.11b)} \\
& \sum_{j\in J} d_j^s x_{ij} - y_i^s \leq u_i^s \quad \forall i \in I,\, s \in S && \text{(3.11c)} \\
& x_{ij}, z_i \in \{0,1\}, \quad \forall i \in I,\, j \in J \\
& y_i^s \geq 0, \quad \forall i \in I,\, s \in S
\end{aligned}
$$

The constraint (3.11a) enables that each customer is served by one location of facility. If no facility is opened at location i, then we cannot serve any customer from this location. This is ensured by (3.11b) and the constraint (3.11c) lets us either to obey the capacity or to pay penalty for unmet demand.

The restricted master problem for the SFLP can be presented as follows.

$$
\begin{aligned}
\min_{\lambda} \quad & \sum_{i\in I}\sum_{k\in K_i'} \hat{c}_i^k \lambda_i^k \\
\text{s.t.} \quad & \sum_{i\in I}\sum_{k\in K_i'} x_{ij}^k \lambda_i^k = 1, \quad j \in J && \text{(3.12a)} \\
& \sum_{k\in K_i'} \lambda_i^k = 1, \quad i \in I && \text{(3.12b)} \\
& \lambda_i^k \in \{0,1\}, \quad k \in K_i',\, i \in I,
\end{aligned}
$$

where $\hat{c_i}^k = f_i z_i^k + \sum_{j\in J} c_{ij} x_{ij}^k + \sum_{s\in S} p^s \rho_i^s y_i^{s,k}$. The feasible assignment $(x_i, y_i) =$

$\left((x_{ij})_{j \in J}, (y_i^s)_{s \in S} \right)$, for location i is generated by the following pricing problem.

$$\begin{aligned} v_i = \min_{x,y} \quad & \sum_{j \in J} (c_{ij} - \pi_j) x_{ij} + \sum_{s \in S} p^s \rho_i^s y_i^s + f_i - \mu_i \\ \text{s.t.} \quad & \sum_{j \in J} d_j^s x_{ij} - y_i^s \le u_i^s, \quad \forall s \in S \\ & x_{ij} \in \{0, 1\}, \quad \forall j \in J \\ & y_i^s \ge 0, \quad \forall s \in S, \end{aligned}$$

where π_j is a dual variable associated with (3.12a) and μ_i is with (3.12b).

The test problems are generated randomly by the following rules (see Silva and Wood [63]): the reference parameters are generated first and each scenario is generated from the reference parameters. Reference customer demands d_j^R are integers from uniform distribution $U(5, 25)$ and reference facility capacities are $u_i^R = 0.8 \sum_{j \in J} d_j^R / |I|$. Demands d_j^s are integers from $\pm 20\%$ of d_j^R and capacities u_i^s are integers from $\pm 10\%$ of u_i^R. To complete parameters, $\rho_i^s = 0.4 \max_{j \in J} c_{ij}^s$ and $p^s = 1/|S|$. The fixed costs are $f_i = 1.5 u_i^R$ and supplying costs c_{ij} are integers from $U(15, 25)$.

Tables 3.1 and 3.3 show the results of column generation for FLP. Silva and Wood [63] reported computational results with the hybrid stabilization, parts of which are copied in Table 3.2 for the comparison purpose where 'min' identifies the result with the fastest result among several instances while 'max' is the one that spent the most. They implemented the algorithm using COIN-OR [35] open solver interface with CPLEX 8.0 on a 2 GHz Pentium 4 processor with 1 GB of RAM, which is very similar specification to one used in this experiment. In the table, the problems with '*' indicated that they were not solved within two hours and the other numbers show the computational time in seconds. We can see that CG results in Tables 3.1 and 3.3 were compatible with the Silva and Wood [63]'s results. The problem indicates the numbers of facility locations, customers and scenarios, respectively. The optimal objective values in the second column of Table 3.1 were obtained by solving the deterministic equivalent problem (3.11) using CPLEX with default relative tolerance 0.01% and two-hour time limit. The last problem 20.60.50 did not find an optimal solution within two hours and z^* in the table was the best solution found within the time limit and it had 0.06% of tolerance gap when CPLEX stopped. The gaps for this problem in the table were based on this value.

The LP relaxation gaps (%) are shown for the DEP (3.11) and the CG formulation (3.12). We see that the CG formulation provided much lower LP relaxation bound than the deterministic equivalent problem. In this computation, CG found optimal solutions for many cases by solving the LP relaxation of the restricted master problem (3.12). In the table, CPU times in seconds and the numbers of columns that were added by solving the pricing problems are displayed, which shows that the hybrid method made CG faster.

Table 3.3 displays the parametric results with the interior point stabilization method. The interior point method solves the auxiliary problem (3.13c) with random objective coefficients at every iteration and the first row of the table shows the number of times that random coefficients were gen-

Table 3.1: Results for Facility Location Problem

		DEP	CG w/o stabilization			CG w/ hybrid method		
		Gap	Gap	Time	#cols	Gap	Time	#cols
Problem	z^*	(%)	(%)	(sec)	(ea)	(%)	(sec)	(ea)
5.15.10	4674	0.74	0.00	5	215	0.00	3	199
5.15.50	3946	0.82	0.00	7	196	0.00	5	224
8.24.50	6501	1.49	0.03	23	521	0.03	15	542
8.48.50	14594	1.05	0.00	139	1372	0.00	55	1121
10.30.50	7707	1.88	0.00	43	782	0.01	23	714
10.40.50	11299	1.39	0.00	91	1186	0.00	43	1001
10.50.50	14951	10.19	0.00	178	1603	0.00	65	1240
20.60.50	*14212*	2.86	0.00	795	2696	0.00	171	2290

Table 3.2: Results for Facility Location Problem in Silva and Wood [63]

	Min (sec)			Max (sec)		
		CG w/o	CG w/		CG w/o	CG w/
Problem	DEP	stab	stab	DEP	stab	stab
5.15.10	0.4	0.6	1.1	1.4	1.1	1.6
5.15.50	1.3	1.4	2.7	3.9	2.7	4.4
8.24.50	6.6	4.2	4.2	5579.0	45.5	29.6
8.48.50	45.7	20.2	14.0	990.2	80.1	114.6
10.30.50	12.6	6.5	6.0	*	26.0	115.0
10.40.50	573.0	14.5	11.3	3559.0	30.8	27.3
10.50.50	*	33.9	27.9	*	112.3	171.7

erated for the problem (3.13c), i.e., one means that the problem was solved twice, with random coefficients (u, u_i) and $(1-u, 1-u_i)$ in turn. The LP relaxation gaps are not displayed here because they showed the same gaps as those of the hybrid stabilization. One pair of random coefficients at each iteration produced the best results among them, which was outperformed by the previous hybrid stabilization.

Table 3.3: Results for the Interior Point Method

	1		2		3		4	
Problem	Time (sec)	#cols (ea)	Time (sec)	#cols (ea)	Time (sec)	#cols (ea)	Time (sec)	#cols (ea)
5.15.10	3	159	4	184	4	158	5	159
5.15.50	7	166	5	169	6	161	6	163
8.24.50	11	378	16	446	22	463	21	441
8.48.50	90	1205	117	1184	161	1228	169	1155
10.30.50	24	660	31	689	37	712	44	739
10.40.50	49	992	74	1063	97	1097	109	1084
10.50.50	99	1357	139	1392	207	1529	210	1387
20.60.50	269	2709	383	2845	499	2875	504	2743

3.2 Generalized Assignment Problem

We begin this section by reformulating (2.13) so that it has exponentially-many columns. The motivation for doing so is that the LP relaxation of (2.13) can give weak lower bounds on z^*, and hence, applying a branch-and-bound algorithm to (2.13) can result in excessive computational effort. After reformulating (2.13), we describe upper and lower bounds on z^* that aid branch-and-bound. Finally, we augment our reformulation of (2.13) to help stabilize

CG iterations within the branch-and-price algorithm (see Section 3.2.3).

3.2.1 Formulation

Let $x_i = (x_{ij})_{j \in J}$ denote the vector of job assignments to agent i, and let x_i^k, $k \in K_i$, index all $2^{|J|}$ such assignments ranging from agent i having no assignments to being assigned all jobs from J. Given assignment x_i^k and scenario $\omega \in \Omega_i$, let $y_i^{\omega k} = \left(\sum_{j \in J} d_{ij}^{\omega} x_{ij}^k - b_i^{\omega}\right)^+$. With $y_i^k = \left(y_i^{\omega k}\right)_{\omega \in \Omega_i}$, we have the pairs (x_i^k, y_i^k) over $k \in K_i$, $i \in I$, representing all feasible solutions to (2.13). Defining $c_i^k = \sum_{j \in J} c_{ij}\, x_{ij}^k + q_i \sum_{\omega \in \Omega_i} p_i^{\omega} y_i^{\omega k}$, i.e., the expected cost of assignment x_i^k, we can reformulate (2.13) as follows

$$
\begin{aligned}
z^* = \min_{\lambda} \quad & \sum_{i \in I} \sum_{k \in K_i} c_i^k \lambda_i^k \\
\text{s.t.} \quad & \sum_{i \in I} \sum_{k \in K_i} x_{ij}^k \lambda_i^k = 1, \quad j \in J && (3.13a) \\
& \sum_{k \in K_i} \lambda_i^k = 1, \quad i \in I && (3.13b) \\
& \lambda_i^k \in \{0,1\}, \quad k \in K_i,\, i \in I. && (3.13c)
\end{aligned}
$$

Constraints (3.13b) and (3.13c) ensure exactly one set of jobs is assigned to each agent and the objective function accounts for the associated expected cost of that assignment. Constraint (3.13a) is then equivalent to (2.13b), i.e., each job is done once. Problem (3.13) is called the full master problem (MP). Of course, it is neither practical nor desirable to explicitly enumerate all feasible assignments of K_i, $i \in I$. Instead we start with modest-sized subsets $K_i' \subset K_i$, $i \in I$, that have the property each job $j \in J$ can be covered by at least one of the agents, i.e., (3.13a)-(3.13c) is feasible when K_i is replaced by K'_i. This leads to the so-called restricted master problem (RMP)

$$
\begin{aligned}
\bar{z} = \min_{\lambda} \quad & \sum_{i \in I} \sum_{k \in K'_i} c_i^k \lambda_i^k \\
\text{s.t.} \quad & \sum_{i \in I} \sum_{k \in K'_i} x_{ij}^k \lambda_i^k = 1, \quad j \in J && (3.14a) \\
& \sum_{k \in K'_i} \lambda_i^k = 1, \quad i \in I && (3.14b) \\
& \lambda_i^k \in \{0,1\}, \quad k \in K'_i,\, i \in I.
\end{aligned}
$$

Let $\hat{\lambda} = \left(\hat{\lambda}_i^k\right)_{k \in K'_i,\, i \in I}$ be a feasible solution to the LP relaxation of (3.14). Then $(\hat{x}_{ij}, \hat{y}_i^{\omega})_{\omega \in \Omega_i, i \in I, j \in J}$ is a feasible solution to the LP relaxation of (2.13), where $\hat{x}_{ij} = \sum_{k \in K'_i} x_{ij}^k \hat{\lambda}_i^k$ and $\hat{y}_i^{\omega} = \sum_{k \in K'_i} y_i^{\omega k} \hat{\lambda}_i^k$. Since the first stage decision is binary, we can have the following relationship between $\hat{\lambda}$ and $\hat{x}$.

Proposition 3.2. *(Savelsbergh [61]) Let $\hat{\lambda} = (\hat{\lambda}_i^k)_{k \in K'_i, i \in I}$ be an optimal solution to the LP relaxation of (3.14). If $\hat{\lambda}_i^k$ is fractional for some i, then there must be a j such that $\hat{x}_{ij} = \sum_{k \in K'_i} x_{ij}^k \hat{\lambda}_i^k$ is fractional.*

Savelsbergh's proof of this proposition is for the deterministic GAP but hinges on the assumption that there are no duplicate columns in the (restricted) master problem. This assumption is valid for our (restricted) master problem, and Savelsbergh's proof carries over directly to our stochastic GAP.

Consider the LP relaxation of (3.14) and let π_j, $j \in J$, and α_i, $i \in I$, be optimal dual variables associated with constraints (3.14a) and (3.14b), respectively. The reduced cost for λ_i^k is then

$$\begin{aligned}\bar{c}_i^k &= c_i^k - \sum_{j\in J} \pi_j x_{ij}^k - \alpha_i \\ &= \sum_{j\in J} (c_{ij} - \pi_j)\, x_{ij}^k + q_i \sum_{\omega\in\Omega} p_i^\omega y_i^{k,\omega} - \alpha_i.\end{aligned}$$

The optimal dual multipliers from the RMP are defined over the problem with columns K_i', $i \in I$. Consider the problem of solving $\min_{k\in K_i'} \bar{c}_i^k$, i.e., finding the column for agent i with the smallest reduced cost over the set K_i'. This column can be found by solving the following pricing problem for agent i

$$\begin{aligned} v_i = \min_{x_i, y_i} \quad & \sum_{j\in J} (c_{ij} - \pi_j) x_{ij} + q_i \sum_{\omega\in\Omega_i} p_i^\omega y_i^\omega \\ \text{s.t.} \quad & \sum_{j\in J} d_{ij}^\omega x_{ij} - y_i^\omega \le b_i^\omega, \quad \omega \in \Omega_i \\ & x_{ij} \in \{0,1\},\ y_i^\omega \ge 0, \quad \omega\in\Omega_i,\ j \in J. \end{aligned} \tag{3.15}$$

Let $(\hat{x}_i, \hat{y}_i)$ solve (3.15), where $\hat{x}_i = (\hat{x}_{ij})_{j\in J}$ and $\hat{y}_i = (\hat{y}_i^\omega)_{\omega\in\Omega_i}$. If $v_i - \alpha_i \equiv \min_{k\in K_i'} \bar{c}_i^k < 0$ then we add a column with $\hat{x}_{ij}$ to the RMP with objective function coefficient $\hat{c}_i = \sum_{j\in J} c_{ij}\hat{x}_{ij} + q_i \sum_{\omega\in\Omega_i} p_i^\omega \hat{y}_i^\omega$, expanding K_i' by one element in the process. We do this, in turn, for each agent, $i \in I$, and then repeat the process by resolving the LP relaxation of the RMP to obtain new dual prices. If $v_i - \alpha_i \ge 0$ for all $i \in I$ then the LP relaxation of the RMP has been solved to optimality. In general, the associated solution will still be fractional. We can apply branch-and-bound to solve the associated integer-constrained RMP. Assuming this RMP is integer-feasible (and it is straight forward to populate K_i' initially with columns that ensure this) the resulting solution is feasible, but not necessarily optimal, to (3.13) or, equivalently, to (2.13). For this reason we will describe a full branch-and-price algorithm in Section 4.

3.2.2 Column Generation Algorithm

The branch-and-price algorithm we describe in the next section is constructed in two parts. The first part obtains an optimal solution to the LP relaxation of a full master program via column generation. Initially, this LP relaxation is that of (3.13) but later is that of (3.13) with some assignment decisions fixed via branching. The column generation part of the algorithm is summarized in Figure 3.2. Step 0 of the CG procedure populates K_i' with an initial set of columns. This can be replaced by a big-M method that we do not detail.

3.2.3 Branch-and-Price

The algorithm described in Figure 3.2 produces a tighter lower bound to (2.13). Since our goal is to find an integer solution, the next step is to apply branch-and-price to the current RMP.

Master Problem In order to carry out column generation at a node in the branch-and-bound tree we must begin with a RMP of the form (3.14), albeit with additional restrictions due to branching, which is feasible. At the root node of the branch-and-bound tree, this can be accomplished in the following

Procedure CG

Step 0 Let $x_i^k = \left(x_{ij}^k\right)_{j \in J}$, $k \in K_i'$, $i \in I$, denote an initial set of possible assignments for each agent. Let $y_i^{\omega k} = \left(\sum_{j \in J} d_{ij}^{\omega} x_{ij}^k - b_i^{\omega}\right)^+$, $\omega \in \Omega_i$, $k \in K_i'$, $i \in I$, and $c_i^k = \sum_{j \in J} c_{ij} x_{ij}^k + q_i \sum_{\omega \in \Omega_i} p_i^{\omega} y_i^{\omega k}$, $k \in K_i'$, $i \in I$.

Initialize RMP (3.14) with columns $\begin{pmatrix} c_i^k \\ x_i^k \end{pmatrix}$, $k \in K_i'$, $i \in I$.

Step 1 Solve the LP relaxation of the RMP (3.14) to obtain optimal value z_{LP}, primal solution λ, and dual solution (π, α).

Step 2 For $i \in I$, solve the pricing problem (3.15) to obtain v_i, $\hat{x}_i$ and $\hat{y}_i$.

Step 3 If $v_i \geq \alpha_i$ for all $i \in I$ stop and report z_{LP} and x^*, where $x_{ij}^* = \sum_{k \in K_i'} x_{ij}^k \lambda_i^k$, $i \in I$, $j \in J$.

Step 4 For $i \in I$, if $v_i < \alpha_i$ then add an element to index set K_i' for new column $\begin{pmatrix} \hat{c}_i \\ \hat{x}_i \end{pmatrix}$, where $\hat{c}_i = \sum_{j \in J} c_{ij} \hat{x}_{ij} + q_i \sum_{\omega \in \Omega_i} p_i^{\omega} \hat{y}_i^{\omega}$.

Goto step 1.

Figure 3.2: A column generation procedure for the GAP.

manner: We partition the set of jobs I among the agents using the assignment costs c_{ij}, $i \in I$, $j \in J$. Specifically, we partition the element of J by forming a column for each agent in which $\hat{x}_{ij} = 1$ if $i \in \arg\min_{i \in I} c_{ij}$ and $\hat{x}_{ij} = 0$ otherwise. To ensure a partition (and the same job is not assigned to two agents) we break ties lexicographically, i.e., if two (or more) agents for job j both have cost $\min_{i \in I} c_{ij}$ the job is assigned to the agent with the smaller index i. The objective function coefficient for agent i's column is computed in the usual manner, $\hat{c}_i = \sum_{j \in J} c_{ij} \hat{x}_{ij} + q_i \sum_{\omega \in \Omega_i} p_i^{\omega} \left(\sum_{j \in J} d_{ij}^{\omega} \hat{x}_{ij} - b_i^{\omega}\right)^+$. After branching restrictions have been added we can again come across an infeasible initial RMP, but of course the associated full MP with branching restrictions is feasible (unless all agents are forbidden to do the same job). So we carry out a similar procedure, if necessary, i.e., we partition unassigned jobs among agents eligible to perform the job. An alternative to this approach is a big-M method in which we form an artificial agent for each job, who can perform only that job at a very high cost.

Branching We consider two branching strategies. The first is the standard approach of branching on a single variable x_{ij} for some i and j. The second branches on a subset of variables in a manner we describe below.

To branch on x_{ij} we form descendent nodes in which we, respectively, fix $x_{ij} = 1$ and $x_{ij} = 0$. We choose for branching the x_{ij} that is closest to $\frac{1}{2}$. Setting $x_{ij} = 1$ implies job j is assigned to agent i while $x_{ij} = 0$ forbids agent i from doing job j. This branching is effected in the λ-variables in the LP relaxation of the RMP (3.14) as follows: To forbid assignment of j to i we set $\lambda_i^k = 0$ for all $k \in K_i'$ with $x_{ij}^k = 1$. To assign j to i we set $\lambda_i^k = 0$ for all $k \in K_i'$ with $x_{ij}^k = 0$ and we set $\lambda_{i'}^k = 0$ for all $i' \neq i$, $k \in K_{i'}'$ with $x_{i'j} = 1$. Of course, setting $\lambda_i^k = 0$ amounts to simply removing the column from the formulation.

Our second strategy replies on the fact that for an arbitrary subset of agents $I' \subseteq I$ and a fixed job $j \in J$ we have the logical constraint $\sum_{i \in I'} x_{ij} \leq 1$ since at most one agent $i \in I'$ will be assigned job j. This allows us to utilize so-called generalized upper bound (GUB) branching: One descendent node has $\sum_{i \in I'} x_{ij} = 0$ and the other has $\sum_{i \in \bar{I}'} x_{ij} = 0$, where $\bar{I}' = I \backslash I'$. We choose I' a priori so that $|I'| \approx \frac{|I|}{2}$ in an attempt to achieve "balance" in the problems for the descendent nodes. We select to do GUB-branching on job j such that $\sum_{i \in I'} x_{ij}$ is closest to $\frac{1}{2}$. Of the two descendent nodes, the one with the most fractional variables is explored first.

Reduced Cost Fixing Reduced-cost fixing is a standard technique in integer programming that allows an integer variable to be fixed to its upper or lower bound based on the value of its reduced cost relative to the optimality gap. Specifically, let $\bar{z}$ be the objective function value of the best candidate solution we have found so far, let $\hat{\lambda} = \left(\hat{\lambda}_i^k\right)_{k \in K_i', i \in I}$ be the optimal solution to the LP relaxation of the RMP at the current node, let $\bar{c} = \left(\bar{c}_i^k\right)_{k \in K_i', i \in I}$ be associated reduced cost, and $\underline{z}$ be the lower bound of Theorem 3.1 at the current node. If $\underline{z} + \bar{c}_i^k \geq \bar{z}$ and $\hat{\lambda}_i^k = 0$ then we can fix $\lambda_i^k = 0$ at this node and its descendents. Similarly, if $\underline{z} - \bar{c}_i^k \leq \bar{z}$ and $\hat{\lambda}_i^k = 1$ then we can fix $\lambda_i^k = 1$. We attempt reduced cost fixing after solving all the pricing problems, i.e., after Step 2 in Figure 3.2, as these allow computing the current value of $\bar{z}$. Figure 3.3 details the reduced-cost fixing procedure.

Fathoming In applying branch-and-bound to solve an integer program (in the absence of branch-and-price), we fathom a node when one of the following conditions is satisfied: (i) the current node is infeasible, (ii) the current node's lower bound exceeds the global upper bound or (iii) the node produces

Subroutine REDUCED-COST-FIXING$(\hat{\lambda}, \pi, \alpha, \underline{z}, \bar{z})$
for $k \in K_i', \ i \in I,$
Let $y_i^{\omega k} = \left(\sum_{j \in J} d_{ij}^{\omega} \, x_{ij}^k - b_i^{\omega}\right)^+$
$\bar{c}_i^k = \sum_{j \in J} (c_{ij} - \pi_j) \, x_{ij}^k + q_i \sum_{\omega \in \Omega_i} p_i^{\omega} y_i^{k\omega} - \alpha_i.$
if $\hat{\lambda}_i^k = 0$ and $\underline{z} + \bar{c}_i^k \geq \bar{z}$ **then** $\lambda_i^k = 0.$
if $\hat{\lambda}_i^k = 1$ and $\underline{z} - \bar{c}_i^k \leq \bar{z}$ **then** $\lambda_i^k = 1.$
end for
return

Figure 3.3: Implementation of reduced-cost fixing

an integer solution. These three rules apply in the branch-and-price setting with the understanding that a node corresponds to the full MP under the given branching restriction. Restated, a rule does not necessarily apply when speaking of the RMP at a node.

We have discussed above that infeasibility of the RMP does not imply infeasibility of the full MP at a node. So, rule (i) rarely applies in our setting. (It only applies if there is a job all agents are forbidden to do due to branching restrictions.) Rule (ii) can be applied to the RMP at a node, i.e., if the lower bound $\underline{z}$ of Theorem 3.1, adapted to an LP relaxation with branching constraints, exceeds $\bar{z}$, the global upper bound, the node can be fathomed. In general we cannot fathom a node when the RMP has an integer solution but has columns with negative reduced costs. Of course, if the objective function value happens to be sufficiently close to the lower bound we can terminate.

3.2.4 Branch-and-Price Algorithm

The following notation is used to describe the the details of the B&P algorithm for the GAP.

P = the LP relaxation of the RMP (3.14).

SP_i = the pricing problem (3.15) for agent i.

$\mathcal{N}$ = the list that contains branching constraints.

$v(C) = 0$ if the current branching constraint C is not explored yet and 1 otherwise.

$\bar{C}$ = the counter branching constraint of C.

BRANCH($\mathcal{N}$) = subroutine that chooses a branching constraint from the list $\mathcal{N}$ and returns its location in the list. The depth-first search returns the last location of the list.

CHOOSEVAR() = subroutine that chooses a branching variable. Since the branching is performed based on the original variables, it returns (i, j) such that $x_{ij} = \sum_{k \in K_i'} x_{ij}^k \lambda_i^k$. which is most fractional for a single variable branching.

The branch-and-price algorithm is summarized in Figure 3.4, where the single variable branching is employed. For the GUB branching, the branching constraint in lines 4 and 25 should be changed to either $C = \{\sum_{i=1}^{p} x_{ij} = 0\}$ or $C = \{\sum_{i=p+1}^{m} x_{ij} = 0\}$ whichever contains the most fractional variable.

Initialization

Perform **Procedure CG** to find z and its solution λ.
Obtain (i, j) = CHOOSEVAR() and add a branching constraint $C = \{x_{ij} = 0\}$ to the list $\mathcal{N}$. If all λ's are integer **then stop**.
Set $v(C) = 0$, $\bar{z} = \infty$, $\underline{z} = -\infty$

Do while $(\mathcal{N} \neq \emptyset)$

k = BRANCH($\mathcal{N}$); $v(C_k) = 1$.
Fix variables in P and SP_i, $i \in I$, according to the branching constraint C_k.

Do while (true)

Solve P to find objective value (z) and primal and dual $(\lambda; \pi, \alpha)$ solutions.
if all λ_i^k's are integer
if $z \leq \bar{z}$ **then**

$$\bar{z} \leftarrow z \text{ and } (\hat{x}, \hat{y}) \leftarrow \left(\left(\sum_{k \in K_i'} x_{ij}^k \lambda_i^k \right)_{i \in I, j \in J}, \left(\sum_{k \in K_i'} y_i^{\omega k} \lambda_i^k \right)_{\omega \in \Omega_i, i \in I} \right).$$

end if

for $i \in I$,
Solve a SP_i with π to obtain its objective value v_i and a solution

$$(x_i, y_i) = \left((x_{ij})_{j \in J}, (y_i^{\omega})_{\omega \in \Omega_i} \right).$$

if $v_i - \alpha_i < 0$ **then** add a column to P with $\begin{pmatrix} \hat{c}_i \\ x_i \end{pmatrix}$ where

$$\hat{c}_i = \sum_{j \in J} c_{ij} x_{ij} + q_i \sum_{\omega \in \Omega_i} p_i^{\omega} y_i^{\omega}.$$

end for
$\underline{z} \leftarrow z + \sum_{i \in I} (v_i - \alpha_i)$.

if $\underline{z} \geq \bar{z}$ **then** do FATHOM($C_k, \mathcal{N}$) and **break**.
if no column is added for all $i \in I$ **then break**.
do REDUCED-COST-FIXING.

End do

Obtain (i, j) = CHOOSEVAR() and add a branching constraint $C = \{x_{ij} = 0\}$ to the list $\mathcal{N}$.
Set $v(C) = 0$.

End do

Report $z^* = \bar{z}$ and $(\hat{x}, \hat{y})$.

Figure 3.4: Implementation of a branch-and-price procedure

The stabilization can be applied to all the nodes in the branch and bound tree but the effect of stabilization is significant only on the root node for the GAP. The subroutine FATHOM() in Figure 3.4 removes the branching constraints that are explored already and is described in Figure 3.5.

Subroutine FATHOM($C_k, \mathcal{N}$)
  Do while (true)
    Restore the bounds for the variables that are fixed according to C_k in P and SP_i, $\forall\, i \in I$.
    $\mathcal{N} \leftarrow \mathcal{N} \backslash \{C_k\}$.
    if $v(C_k) = 0$ **break**
    $k \leftarrow k - 1$.
    if (the list $\mathcal{N}$ is empty) **then return**
  End loop
  Add a constraint $\bar{C}_k$ to the list $\mathcal{N}$.
  return

Figure 3.5: Implementation of fathom

3.2.5 Example

The B&P procedure is explained with a 2-agent, 5-job, 1-scenario problem for illustration. The Table 3.4 contains all input data. Since only one scenario is used, index ω was omitted. For the explaining purpose, a single variable branching is used and the reduced-cost-fixing is not applied.

Table 3.4: Input data for 2-agent, 5-job, 1-scenario example.

i	b_i	d_{i1}	d_{i2}	d_{i3}	d_{i4}	d_{i5}	c_{i1}	c_{i2}	c_{i3}	c_{i4}	c_{i5}	q_i
1	121	94	1	56	67	85	13	112	57	39	20	25
2	85	8	77	64	21	43	110	30	52	81	73	25

Initialization In the beginning, he following feasible solutions are used to obtain the initial RMP.

$$x^1 = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}, \quad y^1 = \begin{pmatrix} 125 \\ 56 \end{pmatrix}, \quad \hat{c}^1 = \begin{pmatrix} 3197 \\ 1482 \end{pmatrix}$$

The dual solution of the LP relaxation of the initial RMP is $\pi = (3197, 1482, 0, 0, 0)$ and $\alpha = (0, 0)$. Now we solve the pricing problems (3.15) with fixed π and α, yielding the following solutions for each i.

$$x^2 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}^T, \quad y^2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \hat{c}^2 = \begin{pmatrix} 125 \\ 140 \end{pmatrix}^T$$

We add x^2 as columns to RMP. The sequence of solving master and pricing problems are repeated until no column is generated by the pricing problems. At the end of the initializing step, the following RMP is obtained.

$$\begin{aligned}
\min \quad & 3197\lambda_1^1 + 125\lambda_1^2 + 795\lambda_1^3 + 283\lambda_1^4 + 714\lambda_1^5 + 834\lambda_1^6 + 3334\lambda_1^7 + 132\lambda_1^8 \\
& \quad +1482\lambda_2^1 + 140\lambda_2^2 + 162\lambda_2^3 + 133\lambda_2^4 + 675\lambda_2^5 + 154\lambda_2^6 + 264\lambda_2^7 + 73\lambda_2^8 \\
\text{s.t.} \quad & \lambda_1^1 + \lambda_1^2 + \lambda_1^3 + \lambda_1^7 + \lambda_2^2 + \lambda_2^3 + \lambda_2^7 = 1 \\
& \lambda_1^2 + \lambda_1^4 + \lambda_1^5 + \lambda_1^7 + \lambda_2^1 + \lambda_2^2 = 1 \\
& \lambda_1^3 + \lambda_1^4 + \lambda_1^5 + \lambda_2^1 + \lambda_2^3 + \lambda_2^4 + \lambda_2^5 = 1 \\
& \lambda_1^1 + \lambda_1^4 + \lambda_1^6 + \lambda_1^7 + \lambda_2^4 + \lambda_2^6 + \lambda_2^7 = 1 \\
& \lambda_1^1 + \lambda_1^5 + \lambda_1^6 + \lambda_1^7 + \lambda_1^8 + \lambda_2^5 + \lambda_2^6 + \lambda_2^7 + \lambda_2^8 = 1 \\
& \lambda_1^1 + \lambda_1^2 + \lambda_1^3 + \lambda_1^4 + \lambda_1^5 + \lambda_1^6 + \lambda_1^7 + \lambda_1^8 = 1 \\
& \lambda_2^1 + \lambda_2^2 + \lambda_2^3 + \lambda_2^4 + \lambda_2^5 + \lambda_2^6 + \lambda_2^7 + \lambda_2^8 = 1 \\
& \lambda_i^j \geq 0, \quad i = 1, 2, \; j = 1, ..., 8
\end{aligned}$$

Branch-and-Price Step The above RMP produces the fraction solution $\lambda_1^4 = \lambda_1^8 = \lambda_2^3 = \lambda_2^7 = 0.5$ and choose the most fractional assignment $\hat{x}_{13} = 0.5$ where $\hat{x}_{13} = \sum_{k \in K_i'} x_{13}^k \lambda_1^k$. Add a branching constraint $\{x_{13} = 0\}$ to the list $\mathcal{N}$; $\mathcal{N} = \{x_{13} = 0\}$. Set $v(\{x_{13} = 0\}) = 0$.

Node 1 Choose the constraint $\{x_{13} = 0\}$ and set $v(\{x_{13} = 0\}) = 1$. To implement the constraint in the RMP, look at the constraint (3.16) and fix the following variables $\lambda_1^3 = \lambda_1^4 = \lambda_1^5 = 0$. Add $\{x_{13} = 0\}$ to all pricing problems.

Solve RMP: The LP relaxation of the RMP produces the objective value of 1880.5 with the primal solution of $\lambda_1^7 = \lambda_1^8 = \lambda_2^3 = \lambda_2^4 = 0.5$ and the dual solution of $\pi = (1615.5, 2935.5, 3107.0, 1586.5, 1622.5)$ and $\alpha = (-4426.0, -4560.5)$.

Solve the pricing problems: The pricing problems produce the objective values $v_1 = -5173.5$ and $v_2 = -7321.0$. Since $v_i < \alpha_i$ for both agents, add the following solutions as columns to RMP.

$$x^9 = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}, \quad y^9 = \begin{pmatrix} 32.0 \\ 128.0 \end{pmatrix}, \quad \hat{c}^9 = \begin{pmatrix} 971 \\ 3546 \end{pmatrix}$$

$$\underline{z} = 1880.5 + (-5173.5 + 4426.0) + (-7321.0 + 4560.5) = -1627.5$$

Solve RMP: Solve the LP relaxation of the RMP yields $z = 1133.0$ with the primal solution of $\lambda_1^9 = 1, \lambda_2^3 = 1$ and the dual solution of $\pi = (1721.3, 137.0, 3145.0, 1012.7, 1554.7)$ and $\alpha = (-1733.3, -4024.7)$. Since we have an integer solution, update $\bar{z}$; $\bar{z} = 1133.0$.

Solve the pricing problems: The pricing problems yield the objective value $v_1 = -1793.0$ and $v_2 = -5842.7$. Since $v_i < \alpha_i$ for both agents, add the following solutions as columns to the RMP.

$$x^{10} = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{pmatrix}, \quad y^{10} = \begin{pmatrix} 59.0 \\ 51.0 \end{pmatrix}, \quad \hat{c}^{10} = \begin{pmatrix} 1620 \\ 1591 \end{pmatrix}$$

$$\underline{z} = 1133.0 + (-1793.0 + 1733.3) + (-5842.7 + 4024.7) = -744.7$$

Solve RMP: The LP relaxation of the RMP yields $z = 990.5$ with the primal solution of $\lambda_1^2 = \lambda_1^8 = \lambda_2^4 = \lambda_2^{10} = 0.5$ and the dual solution of $\pi = (725.5, -5.5, 1327.0, 696.5, 732.5)$ and $\alpha = (-595.0, -1890.5)$.

Solve the pricing problems: The pricing problems have the objective value $v_1 = -712.5$ and $v_2 = -2306.0$. Add the following solutions as columns to RMP.

$$x^{11} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix}, \quad y^{11} = \begin{pmatrix} 0.0 \\ 8.0 \end{pmatrix}, \quad \hat{c}^{11} = \begin{pmatrix} 13 \\ 443 \end{pmatrix}$$

$$\underline{z} = 990.5 + (-712.5 + 595.0) + (-2306.0 + 1890.5) = 457.5$$

Solve RMP: The LP relaxation of the RMP yields $z = 575.0$ with the primal solution of $\lambda_1^8 = 1$, $\lambda_2^{11} = 1$ and the dual solution of $\pi = (725.5, 112.0, 1327.0, 696.5, 732.5)$ and $\alpha = (-712.5, -1890.5)$. Since we have an integer solution and $z < \bar{z}$, update $\bar{z} = 575.0$.

Solve the pricing problems: The pricing problems produce the objective value $v_1 = -712.5$ and $v_2 = -2306.0$. Add the solution from the second pricing problem as a column to RMP.

$$x_2^{12} = (1, 0, 1, 1, 0), \quad y_2^{12} = 8.0, \quad \hat{c}_2^{12} = 443$$

$$\underline{z} = 575.0 + (-712.5 + 712.5) + (-2306.0 + 1890.5) = 159.5$$

Solve RMP: The LP relaxation of the RMP yields $z = 575.0$ with the primal solution of $\lambda_1^8 = 1$, $\lambda_2^{11} = 1$ and the dual solution of $\pi = (310.0, 112.0, 496.0, 281.0, 317.0)$ and $\alpha = (-297.0, -644.0)$.

Solve the pricing problems: The pricing problems produce the objective value $v_1 = -297.0$ and $v_2 = -644.0$. Since $v_i - \alpha_i = 0$ for both agents and the RMP has an integer solution, we fathom the node; remove $\lambda_1^3 = \lambda_1^4 = \lambda_1^5 = 0$ from the RMP and $x_{13} = 0$ from the pricing problems. Remove $\{x_{13} = 0\}$ from the list $\mathcal{N}$ and add $\{x_{13} = 1\}$ to the list $\mathcal{N}$ and set $v(\{x_{13} = 1\}) = 0$.

Node 2 Choose the constraint $\{x_{13} = 1\}$ and set $v(\{x_{13} = 1\}) = 1$. Fix the following variables $\lambda_1^3 = \lambda_1^4 = \lambda_1^5 = 0$ of the RMP and fix $x_{13} = 1$ of pricing problems.

Solve RMP: The LP relaxation of the RMP yields 704.0 with the primal solution of $\lambda_1^3 = 0.25$, $\lambda_1^4 = 0.5$, $\lambda_1^5 = 0.25$, $\lambda_2^2 = 0.25, \lambda_2^7 = 0.5$, $\lambda_2^8 = 0.25$ and the dual solution of $\pi = (348.0, -7.0, 1690.0, 157.0, 274.0)$ and $\alpha = (-1243.0, -201.0)$.

Solve the pricing problems: The pricing problems produce the objective value $v_1 = -1633.0$ and $v_2 = -439.0$. Since $v_i < \alpha_i$ for both agents, add the following solutions as columns to the RMP.

$$x_1^{12} = (0, 0, 1, 0, 0), \quad y_1^{12} = 0.0, \quad \hat{c}_1^{12} = 57$$

$$x_2^{13} = (1, 0, 0, 0, 1), \quad y_2^{13} = 0.0, \quad \hat{c}_2^{13} = 183$$

$$\underline{z} = 704.0 + (-1633.0 + 1243.0) + (-439.0 + 201.5) = 76.0$$

Solve RMP: The LP relaxation of the RMP yields $z = 466.0$ with the primal solution of $\lambda_1^4 = 1, \lambda_2^{13} = 1$ and the dual solution of $\pi = (110.0, 226.0, 1452.0, 0.0, 269.0)$ and $\alpha = (-1395.0, -196.0)$. Update $\bar{z} = 466$.

Solve the pricing problems: The pricing problems produce the objective value $v_1 = -1509.0$ and $v_2 = -196.0$. Add the solution from the first pricing problem as a column to the RMP.

$$x_1^{13} = (0, 1, 1, 0, 0), \quad y_1^{13} = 0.0, \quad \hat{c}_1^{13} = 169$$

$$\underline{z} = 575.0 + (-1509.0 + 1395.0) + (-196.0 + 196.0) = 352.0$$

Solve RMP: The LP relaxation of the RMP yields $z = 433.0$ with the primal solution of $\lambda_1^{13} = 1$, $\lambda_2^7 = 1$ and the dual solution of $\pi = (110.0, 112.0, 1452.0, 114.0, 122.0)$ and $\alpha = (-1395.0, -82.0)$. Update $\bar{z} = 433$.

Solve the pricing problems: The pricing problems produce the objective value $v_1 = -1420.0$ and $v_2 = -82.0$. Add the solution from the first pricing problem as a column to the RMP.

$$x_1^{14} = (0, 0, 1, 1, 0), \quad y_1^{14} = 2.0, \quad \hat{c}_1^{14} = 146$$

$$\underline{z} = 433.0 + (-1420.0 + 1395.0) + (-82.0 + 82.0) = 408.0$$

Solve the RMP: The LP relaxation of the RMP yields $z = 433.0$ with the primal solution of $\lambda_1^{13} = 1$, $\lambda_2^7 = 1$ and the dual solution of $\pi = (110.0, 137.0, 1452.0, 114.0, 147.0)$ and $\alpha = (-1420.0, -107.0)$. Update $\bar{z} = 433$.

Solve the pricing problems: The pricing problems produce the objective value $v_1 = -1420.0$ and $v_2 = -107.0$. Since $v_i - \alpha_i = 0$ for both agents, we fathom the node; remove $\lambda_1^3 = \lambda_1^4 = \lambda_1^5 = 0$ from the RMP and $x_{13} = 1$ from the pricing problems. Remove $\{x_{13} = 1\}$ from the list $\mathcal{N}$. Since the list $\mathcal{N}$ is empty, we terminate the algorithm.

3.2.6 Computational Results for GAP

The branch-and-price algorithm was implemented using JAVA with CPLEX 9.0 and its performance was compared with the deterministic equivalent problems. All computations were performed on a PC with a dual 1.8GHz CPU, 1GB memory, running SuSE Linux.

The similar setting was used for this experiment as in Section 2.2.2; the same data generating rules and the same number of jobs and agents were used ($m = 5, 10, 20$ and $n = 30, 50$). The more scenarios were used to show the performance of branch-and-price algorithm.

The pricing problems were solved as integer problems using CPLEX with the default value of relative tolerance, which is 0.0001. In an early stage of CG procedure, two approaches were examined to make the master problem feasible; (i) big-M method and (ii) using an initial solution. For this experiment, only root node were solved.

The result in Table 3.5 shows the computational time of B&P using initial solutions and that with a big-M method. Initial solutions, even though their quality were not good, performed better than a big-M method and this was because the algorithm spent time to find a feasible solution. For the remainder of this section, initial solution method was used.

Table 3.5: Comparison of using an initial solution and big-M method (seconds)

Problem	With initial solution	Using big-M method
A. 5.30.10	9.4	10.7
A. 5.50.10	68.5	52.8
A.10.30.10	9.4	16.0
A.10.50.10	41.1	72.6
A.20.30.10	8.1	44.8
A.20.50.10	27.2	140.1
B. 5.30.10	14.2	18.2
B. 5.50.10	80.0	144.9
B.10.30.10	7.2	21.0
B.10.50.10	32.2	140.0
B.20.30.10	5.4	42.1
B.20.50.10	19.0	144.5
C. 5.30.10	14.8	16.3
C. 5.50.10	69.1	90.8
C.10.30.10	6.8	20.9
C.10.50.10	33.8	89.9
C.20.30.10	8.6	45.0
C.20.50.10	24.2	173.8
D. 5.30.10	14.5	25.7
D. 5.50.10	123.9	174.6
D.10.30.10	8.2	21.4
D.10.50.10	42.7	105.8
D.20.30.10	10.3	43.9
D.20.50.10	26.6	260.1

In the next experiment, the performance of branching strategies was examined. In this experiment, fathoming by optimality was applied when no column was added and the reduced-cost-fixing was not applied. Tables 3.6 includes the results of two branching strategies without stabilization. In the table, the first column identifies the name of problems, the second column displays the objective function values and the remainder of the table has the CPU times in seconds, the number of nodes explored in a branch-and-bound tree and the number of columns added during the column generation phase for both strategies. It shows that the GUB branching strategy outperformed the single variable branching method, especially when the number of branch-and-bound nodes was large except one problem D.10.50.10. Note that some problems were solved in column generation procedure, i.e., the number of branch-and-bound nodes was 0, and their computational times should read equivalently, i.e., B.10.50.10 was solved in 36.1 second with a GUB-branching and 34.8 seconds with a single variable branching but both measured the same calculation. Hence the GUB-branching method was used in the following experiments.

The results with stabilization are contained in Table 3.7 where the GUB branching was used. For stabilization, ϵ_j^+ and ϵ_j^- were set to 0.1 initially and decreased by the rate of 0.1 when no column was added. When ϵ_j was less than 0.001, it was set to zero. The current dual solutions were used for κ_j^+ and κ_j^- and they were updated when no column was added. Updating them at every iteration would be an alternate but from the empirical result we decided to update them when no column was added. Comparing this table with the GUB branching results in Table 3.6, we can see that the stabilization helped the problems to be solved faster. Stabilization improved the computational time by 65% on average.

The next computation is for the effect of the reduced cost fixing. Table 3.8 shows the results with the reduced cost fixing, where A and B refer two optimal-fathom methods; A has the results that the rule (iii) was applied when no column was generated while B is for the case that the rule was applied immediate after the RMP when the RMP produced an integer solution. A and B didn't show much difference but A was slightly better for the most time consumption problem, D.5.50.10. Compared with Table 3.7, both reduced cost fixing methods improved the computational time especially when the number of branch-and-bound nodes was large. The reduced cost fixing was employed in the following experiments.

In the final computations, the quality of the algorithm is tested. To verify the quality, the deterministic equivalent problems (DEP) (2.13) were implemented and two sets of data were used for the solution quality; one was the same data used in the previous computations and the other had various numbers of scenarios.

In Table 3.9, the results for the LP gaps and computational times for the whole branch-and-price algorithm are displayed, where the column LP has the values of integrality gap in percent of the original problem (2.13) and the column LP_{CG} shows the gaps of the column generation formulation (3.14).

Table 3.6: Comparison of branching strategies without stabilization

Problem	z	GUB-branching Time (sec)	num nodes	Opt node	num cols	Single variable-branching Time (sec)	num nodes	Opt node	num cols
A. 5.30.10	363	7.9	2	1	513	14.1	2	1	509
A. 5.50.10	620	120.8	16	6	2196	216.3	10	3	1945
A.10.30.10	327	7.5	2	1	491	13.4	2	1	489
A.10.50.10	557	26.3	0	0	1069	26.1	0	0	1069
A.20.30.10	309	6.4	0	0	459	6.5	0	0	4593
A.20.50.10	513	30.1	0	0	1269	30.1	0	0	1269
B. 5.30.10	333	9.9	2	2	557	12.0	2	2	557
B. 5.50.10	611	80.9	6	5	1742	138.5	4	4	1616
B.10.30.10	328	7.0	2	2	450	8.5	2	2	450
B.10.50.10	551	36.1	0	0	1292	34.8	0	0	1292
B.20.30.10	307	9.5	4	3	575	9.8	6	4	578
B.20.50.10	516	22.8	2	1	1008	30.6	2	1	1008
C. 5.30.10	393	38.0	42	39	862	36.7	24	21	669
C. 5.50.10	599	334.4	40	40	3141	1707.9	152	152	5291
C.10.30.10	361	9.3	2	1	515	10.8	2	1	514
C.10.50.10	575	30.9	0	0	1124	30.3	0	0	1124
C.20.30.10	345	11.0	2	2	655	14.5	2	2	655
C.20.50.10	561	35.7	0	0	1328	35.3	0	0	1328
D. 5.30.10	3073	35.6	26	5	889	53.3	60	36	1138
D. 5.50.10	5063	466.3	88	58	3626	794.3	142	125	4984
D.10.30.10	3037	15.5	0	0	788	15.5	0	0	788
D.10.50.10	4990	173.3	52	51	2292	71.8	4	2	1754
D.20.30.10	3016	30.3	0	0	659	30.5	0	0	1308
D.20.50.10	4972	127.7	10	10	2676	133.4	14	14	2725

Table 3.7: The results with stabilization

Problem	z	Time (sec)	num nodes	Optimal node	num cols
A. 5.30.10	363	6.4	2	1	372
A. 5.50.10	620	53.6	12	5	1343
A.10.30.10	327	6.5	2	2	363
A.10.50.10	557	16.1	2	1	662
A.20.30.10	309	5.5	0	0	291
A.20.50.10	513	14.9	0	0	691
B. 5.30.10	333	6.4	0	0	351
B. 5.50.10	611	67.9	14	3	1280
B.10.30.10	328	5.6	0	0	335
B.10.50.10	551	16.4	0	0	700
B.20.30.10	307	6.9	2	1	359
B.20.50.10	516	12.9	0	0	612
C. 5.30.10	393	22.5	26	7	521
C. 5.50.10	599	57.5	2	1	989
C.10.30.10	361	7.0	2	1	319
C.10.50.10	575	20.4	0	0	706
C.20.30.10	345	14.2	12	12	489
C.20.50.10	561	18.0	0	0	740
D. 5.30.10	3073	32.5	24	16	709
D. 5.50.10	5063	151.5	22	2	1508
D.10.30.10	3037	9.6	0	0	448
D.10.50.10	4990	137.7	114	113	1845
D.20.30.10	3016	11.6	0	0	589
D.20.50.10	4972	54.1	18	18	1112

Table 3.8: The results with reduced cost fixing

		A				B			
Problem	z	Time (sec)	num nodes	Opt node	num cols	Time (sec)	num nodes	Opt node	num cols
A. 5.30.10	363	5.9	2	1	357	6.0	4	1	348
A. 5.50.10	620	47.5	14	6	1172	41.4	10	10	1105
A.10.30.10	327	6.9	2	2	401	7.0	2	2	397
A.10.50.10	557	15.9	2	1	671	16.0	2	1	662
A.20.30.10	309	5.1	0	0	291	5.5	0	0	291
A.20.50.10	513	14.4	0	0	691	15.2	0	0	691
B. 5.30.10	333	6.5	0	0	351	6.4	0	0	351
B. 5.50.10	611	39.0	12	2	1066	36.2	8	2	988
B.10.30.10	328	5.4	0	0	335	5.7	0	0	335
B.10.50.10	551	16.1	0	0	700	16.8	0	0	700
B.20.30.10	307	6.7	2	1	360	6.7	2	1	359
B.20.50.10	516	13.1	0	0	612	13.1	0	0	612
C. 5.30.10	393	30.7	6	5	483	13.7	42	42	742
C. 5.50.10	599	45.7	2	1	1019	45.7	2	1	1047
C.10.30.10	361	7.5	2	1	319	6.6	2	2	339
C.10.50.10	575	19.8	0	0	706	20.9	0	0	706
C.20.30.10	345	16.5	10	10	535	14.6	16	16	538
C.20.50.10	561	17.5	0	0	740	18.4	0	0	740
D. 5.30.10	3073	30.2	30	27	629	33.8	28	5	759
D. 5.50.10	5063	131.9	24	2	1910	166.3	30	5	1701
D.10.30.10	3037	9.2	0	0	448	10.6	0	0	448
D.10.50.10	4990	66.4	14	14	1208	60.1	16	16	1291
D.20.30.10	3016	11.3	0	0	589	12.0	0	0	589
D.20.50.10	4972	37.5	6	6	1063	35.2	6	6	1087

The column generation procedure provided much smaller gaps. Note that there were some cases that the B&P produced no LP gap, i.e., the problems in class A. The DEPs of two problems, C.20.50.10 and D.20.50.10, were not solved in two hours.

In the following computations, 10 instances were generated for each problem and the average computational times were reported. The data in the smaller number of scenarios were reused in the larger number of scenarios, i.e., A.20.60.20 included A.20.60.10 data and 10 more scenarios. The various numbers of scenarios were considered and the results are found in Tables 3.10 and 3.11. The scenarios of data in the Table 3.10 were generated from a uniform distribution of ±20%, i.e., $d_{ij} \pm 20\%$ and $b_j \pm 20\%$, while ±50% were used for those in Table 3.11. In this computations, the GUB branching, stabilization and reduced-cost-fixing methods were employed. For the reduced-cost-fixing, method A was applied to avoid the worst time consumption.

In both tables, the first column of DEP shows the number of instances that spent two full hours by CPLEX out of ten and those instances were excluded from the average computational time, i.e., the reported time is the averaged one for those solved less than 2 hours. Note that DEP had many problems that some of the instances were not solved in two hours but B&P solved all instances within minutes. It is common in the results that the more distributed data took more time except the class D, which spent more time in solving data with less variations. It seems that adding more variability made it easier to solve class D problems.

Table 3.9: Solution quality: LP gaps and computational times

Problem	z	Gap (%)		Time (sec)	
		LP	LP_{CG}	DEP	B&P
A. 5.30.10	363.0	0.25	0.00	0.7	6.4
A. 5.50.10	620.0	0.25	0.00	0.9	53.6
A.10.30.10	327.0	0.77	0.00	0.8	6.5
A.10.50.10	557.0	0.60	0.00	1.0	16.1
A.20.30.10	309.0	0.32	0.00	0.9	5.5
A.20.50.10	513.0	0.11	0.00	1.0	14.9
B. 5.30.10	333.0	0.24	0.00	0.8	6.4
B. 5.50.10	611.0	0.34	0.04	0.9	51.9
B.10.30.10	328.0	1.00	0.00	0.8	5.6
B.10.50.10	551.0	0.00	0.00	0.9	16.7
B.20.30.10	307.0	0.95	0.00	0.9	6.6
B.20.50.10	516.0	0.00	0.00	0.9	12.9
C. 5.30.10	392.7	2.11	0.29	3.3	18.5
C. 5.50.10	599.0	1.05	0.02	3.1	45.5
C.10.30.10	361.0	4.53	0.08	212.9	6.7
C.10.50.10	575.2	1.67	0.00	94.4	20.8
C.20.30.10	345.3	7.33	0.06	46.1	12.6
C.20.50.10	561.0	2.68	0.00	-	18.3
D. 5.30.10	3073.0	0.54	0.11	3.1	28.7
D. 5.50.10	5063.0	0.21	0.04	38.2	108.5
D.10.30.10	3037.1	1.18	0.00	3.1	9.2
D.10.50.10	4989.6	0.37	0.01	1058.4	179.9
D.20.30.10	3016.0	1.60	0.00	1.1	11.1
D.20.50.10	4973.0	0.89	0.02	-	48.2

Table 3.10: Computational time with various scenarios (±20%)

Problem	DEP		B&P		
	# instances	Time (sec)	Time (sec)	# nodes	# cols
A.20.60.10	1	0.4	37.6	6	602
A.20.60.20	0	0.6	46.7	10	642
A.20.60.30	0	0.8	58.3	9	755
A.20.60.40	0	0.9	58.1	6	833
A.20.60.50	0	1.2	61.6	4	657
A.20.60.60	0	1.4	71.4	7	468
A.20.60.70	0	1.6	70.0	4	682
A.20.60.80	0	1.7	80.1	4	695
A.20.60.90	0	1.8	90.9	8	621
A.20.60.100	1	2.0	88.7	5	706
B.20.50.10	0	82.3	33.8	3	452
B.20.50.20	1	146.0	54.4	16	423
B.20.50.30	0	505.2	48.3	2	340
B.20.50.40	1	175.1	50.8	2	479
B.20.50.50	0	40.4	49.4	2	463
B.20.50.60	1	49.2	53.7	2	420
B.20.50.70	0	111.0	59.8	2	503
B.20.50.80	1	138.0	78.2	3	434
B.20.50.90	1	222.0	108.0	3	430
B.20.50.100	0	94.6	99.4	3	385
C.20.30.10	1	1221.3	39.6	3	204
C.20.30.20	0	1573.8	73.8	2	153
C.20.30.30	0	1364.8	80.2	3	274
C.20.30.40	0	1747.5	76.6	2	107
C.20.30.50	0	1897.1	85.9	3	223
C.20.30.60	0	1888.3	81.3	1	263
C.20.30.70	0	2163.7	86.9	2	262
C.20.30.80	0	2128.0	93.2	2	171
C.20.30.90	0	2670.6	102.8	3	220
C.20.30.100	1	3324.3	102.5	3	220
D.20.30.10	1	505.9	45.9	11	634
D.20.30.20	0	577.2	68.3	19	544
D.20.30.30	0	570.3	66.8	14	654
D.20.30.40	0	1079.6	72.9	10	578
D.20.30.50	0	1594.9	98.1	21	731
D.20.30.60	0	1809.5	92.6	15	442
D.20.30.70	0	2092.0	87.3	8	615
D.20.30.80	0	2713.8	86.5	6	487
D.20.30.90	0	3744.8	117.9	11	507
D.20.30.100	1	3499.6	120.3	9	572

Table 3.11: Computational time with various scenarios (±50%)

Problem	DEP # instances	DEP Time (sec)	B&P Time (sec)	B&P # nodes	B&P # cols
A.20.60.10	1	1.0	37.9	3	704
A.20.60.20	0	14.8	45.0	2	377
A.20.60.30	1	837.6	63.6	6	797
A.20.60.40	0	476.1	68.9	5	419
A.20.60.50	1	822.7	79.9	6	590
A.20.60.60	1	1084.7	91.8	6	718
A.20.60.70	1	1516.0	101.9	3	496
A.20.60.80	2	2006.9	107.9	5	623
A.20.60.90	2	1973.2	158.1	14	620
A.20.60.100	2	1968.5	122.9	5	640
B.20.50.10	0	67.3	26.4	1	242
B.20.50.20	3	2198.3	40.5	8	486
B.20.50.30	3	2259.3	51.0	9	598
B.20.50.40	3	2613.0	51.9	4	423
B.20.50.50	3	2902.9	72.4	12	506
B.20.50.60	5	4299.7	72.3	10	608
B.20.50.70	3	3373.4	58.9	2	381
B.20.50.80	4	3314.8	64.6	2	317
B.20.50.90	2	2892.4	81.3	4	438
B.20.50.100	3	3117.5	100.3	8	518
C.20.30.10	10	7249.7	41.2	6	407
C.20.30.20	6	5663.7	75.9	2	119
C.20.30.30	8	6524.3	74.3	2	192
C.20.30.40	9	7038.3	79.1	1	116
C.20.30.50	9	6938.0	80.1	1	107
C.20.30.60	7	6527.4	86.8	1	149
C.20.30.70	9	7012.0	88.2	1	151
C.20.30.80	9	7093.1	88.3	1	139
C.20.30.90	10	7213.9	98.3	1	172
C.20.30.100	10	7222.9	99.5	2	208
D.20.30.10	0	111.8	26.9	3	280
D.20.30.20	0	22.0	30.7	3	268
D.20.30.30	0	13.2	40.0	9	433
D.20.30.40	0	20.7	34.9	2	391
D.20.30.50	0	10.3	39.8	3	461
D.20.30.60	0	15.3	46.3	5	336
D.20.30.70	0	17.7	52.0	5	270
D.20.30.80	0	16.6	49.3	2	268
D.20.30.90	0	19.1	48.9	1	320
D.20.30.100	0	19.0	61.1	6	399

3.3 Shift Planning And Scheduling Problem

As reviewed in the previous section, column generation performed well if its slow convergency can be improved and stabilization worked for a stochastic facility location problem and a stochastic generalized assignment problem. In this section, column generation is applied to a shift planning and scheduling problem for which the model is established in Section 2.3.

3.3.1 Formulation

One of the most common ways of decomposition for a stochastic programming problem is to decompose it according to its scenarios and, for the column generation formulation, the problem is transformed into the one with nonanticipativity constraints. If we put those constraints into the master problem, then each pricing problem becomes a one-scenario problem. Assuming that the number of scenarios is finite, let's duplicate the first stage variables w, v and x in scenario, i.e., $w_a \rightarrow w_a^\omega, \omega \in \Omega, \forall a$, and add the following nonanticipativity constraint.

$$w_a^0 = w_a^\omega, \quad \omega \in \Omega, \forall a.$$

The (restricted) master problem can be presented as follows.

$$
\begin{aligned}
\min_{w^0,v^0,x^0,\lambda} \quad & \sum_{\omega\in\Omega}\sum_{k\in K'^{\omega}} \hat{c}_k^{\omega}\lambda_k^{\omega} \\
\text{s.t.} \quad & w_a^0 - \sum_{k\in K'^{\omega}} w_a^{\omega,k}\lambda_k^{\omega} = d, \quad \forall a,\omega && (3.16a)\\
& v_b^0 - \sum_{k\in K'^{\omega}} v_b^{\omega,k}\lambda_k^{\omega} = d, \quad \forall b,\omega && (3.16b)\\
& x_{fd}^0 - \sum_{k\in K'^{\omega}} x_{fd}^{\omega,k}\lambda_k^{\omega} = d, \quad \forall f,d,\omega && (3.16c)\\
& \sum_{k\in K'^{\omega}} \lambda_k^{\omega} = 1, \quad \forall\omega && (3.16d)\\
& \lambda_k^{\omega} \geq 0, \quad k\in K'^{\omega}, \forall\omega \\
& (w_a^0, v_b^0, x_{fd}^0) \in Z, \quad \forall a,b,f,d,
\end{aligned}
$$

where $\hat{c}_k^{\omega}$ is the scheduling cost with $(w_a^{\omega,k}, v_b^{\omega,k}, x_{fd}^{\omega,k})$ and can be calculated by

$$
\begin{aligned}
\hat{c}_k^{\omega} \;=\; & \sum_{a=1}^{n^A} c_a^F w_a^{\omega,k} + \sum_{d=1}^{7}\sum_{p=1}^{n^P} c_p^P y_{pd}^{\omega,k} + \sum_{d=1}^{7}\sum_{f=1}^{n^F} c_f^{OT1} {\mu_{fd}^1}^{\omega,k} + \sum_{d=1}^{7}\sum_{f=1}^{n^F} c_f^{OT2} {\mu_{fd}^2}^{\omega,k} \\
& + \sum_{d=1}^{7}\sum_{p=1}^{n^P} c_p^C \gamma_{pd}^{\omega,k} + \sum_{a=1}^{n^A} c^U {w_a^U}^{\omega,k}
\end{aligned}
$$

Based on the reduced cost for λ^{ω}, the following pricing problem, for each scenario ω, will be solved to generate a column.

$$
\begin{aligned}
v^{\omega} = \min_{w,v,x,y,\mu^1,\mu^2,\gamma,w^U} \quad & \sum_{a=1}^{n^A} (p^{\omega} c_a^F + \pi_a^{W,\omega}) w_a^{\omega} + \sum_{b=1}^{n^B} \pi_b^{V,\omega} v_b^{\omega} + \sum_{d=1}^{7} \sum_{f=1}^{n^F} \pi_{fd}^{X,\omega} x_{fd}^{\omega} \\
& + \sum_{d=1}^{7} \sum_{p=1}^{n^P} p^{\omega} c_p^P y_{pd}^{\omega} + \sum_{d=1}^{7} \sum_{f=1}^{n^F} p^{\omega} (c_f^{OT1} {\mu_{fd}^1}^{\omega} + c_f^{OT2} {\mu_{fd}^2}^{\omega}) \\
& + \sum_{d=1}^{7} \sum_{p=1}^{n^P} p^{\omega} c_p^C \gamma_{pd}^{\omega} + \sum_{a=1}^{n^A} p^{\omega} c^U w_a^{U\omega} \\
\text{s.t.} \quad & \sum_{a=1}^{n^A} w_a^{\omega} - \rho \sum_{b=1}^{n^B} v_b^{\omega} \geq 0 \\
& w_a^{\omega} - \frac{1}{5} \sum_{d=1}^{7} \sum_{f=1}^{n^F} A_{af} x_{fd}^{\omega} \geq 0, \quad \forall a \\
& w_a^{\omega} - \sum_{f=1}^{n^F} A_{af} x_{fd}^{\omega} \geq 0, \quad \forall a, d \\
& w_a^{\omega}, v_b^{\omega}, x_{fd}^{\omega} \in Z_+ \\
& (2.20b) - (2.20r)
\end{aligned}
$$

where $\pi_a^{W,\omega}, \pi_b^{V,\omega}$, and $\pi_{fd}^{X,\omega}$ are dual variables associated with $(3.16a), (3.16b)$, and $(3.16c)$, respectively. Let α^{ω} denote a dual variable associated with (3.16d) and if $v^{\omega} \geq \alpha^{\omega}$ for all ω, we stop the algorithm and apply branch-and-bound. Otherwise we add a column to the master problem with a solution $(w_a^{\omega}, v_b^{\omega}, x_{fd}^{\omega})$.

3.3.2 Computational Results for SPSP

To test the performance of column generation, a reduced size of data from those used in Section 2.5 was used. Only one day schedule was used with two scenarios and for daily demand, Monday profile was used with high and low demands. The other parameters were set to the same as those in the previous section; the number of periods in a day is $n^T = 24$, the number of full-time shifts is $n^F = 24$, two types of part-time shifts, i.e., 4 hour shift and 5 hour shift, the number of part-time shifts is $n^P = 48$ and both part-time shifts do not have a break. For the daily profile, Monday profile was used. For the feasibility of the master problem, big-M method was used. To measure the quality of solution, the deterministic equivalent problem was solved with 1% of the relative tolerance gap. For test purpose, only the root node of branch-and-bound procedure was solved in this section.

Table 3.12 shows the results for column generation with various stabilization methods. The first column shows the type of the stabilization method; "None" is for a column generation method without stabilization, "H" stands for the one with the hybrid stabilization method and the remainders are for those with the interior point method where the number shows how many random coefficients were generated, i.e., 1 means that one instance of random objective coefficients for u and u_i was generated and the dual problem (3.4) was solved twice with (u, u_i) and $(1 - u, 1 - u_i)$ to produce the dual solution at every iteration. The second column is the computational time and the last column presents for the number of columns added by solving pricing problems. For the hybrid method, κ^+ and κ^- were updated with the current dual solution at every iteration and ϵ^+ and ϵ^- were reduced by the rate of 0.1 when no column was added. Its initial value was 1 and was set to 0 when it was less than 10^4. We can see that the hybrid method outperformed the interior point

Table 3.12: Parametric results for stabilization

Method	Gap (%)	Time (sec)	# cols
None	0.05	4914	5295
H	0.07	170	294
1	0.04	776	1141
2	0.03	805	1065
3	0.53	931	1136

method and especially its CPU time was only 3% of the column generation without stabilization. Hence the hybrid stabilization method was used for the later computations.

In the next experiment, CG performance was examined according to the increasing number of days. Table 3.13 includes the LP relaxation gaps, computational time and the number of columns added for column generation approach. "LP Gap" is the LP relaxation gap for the deterministic equivalent form of the model (2.19)-(2.20) and "LP_{CG} Gap" is for (3.16) in percent. All problems were solved with two scenarios. The CG formulation (3.16) provided tighter bounds than the deterministic equivalent problem of (2.19)-(2.20). The computational time grew quickly as the number of days increased and this is because increasing number of days brought larger numbers of nonanticipativity constraints in the master problem, which makes the master problem hard to be improved with feasible solutions.

The last experiment examines how CG performs as the number of scenarios grows for one day schedule. Table 3.14 includes the similar results as in Table 3.13. The computational time grew as the number of scenarios became

Table 3.13: Parametric results I for column generation

# days	z^*	LP Gap (%)	Column Generation LP_{CG} Gap (%)	Time (sec)	# cols
1	120254	7.64	0.07	170	271
2	128777	11.63	3.98	191	528
3	121128	0.71	0.10	706	645
4	126449	0.87	0.46	1052	645
5	131954	0.90	0.30	2716	862

large.

Table 3.14: Parametric results II for column generation

# scenarios	z^*	LP Gap (%)	Column Generation LP_{CG} Gap (%)	Time (sec)	# cols
2	120254	7.64	0.07	170	294
3	120058	7.89	0.17	302	673
4	120201	7.92	0.21	560	1124
5	120147	7.85	0.17	1085	1619

The 7-day problems that used in Section 2.5 were not able to be solved by CG. It seems that the dimension of the nonanticipativity constraints is too large to solve.

Chapter 4

Summary and Conclusions

As companies involved in the service industry become more sensitive to the needs of their customers, it is increasingly important for them to take demand uncertainty into account at all levels of planning. One way to do this is with the use of stochastic models. This dissertation considered two applications in which two-stage stochastic integer programs were used to address uncertainty. The first was a generalized assignment problem, a typical combinatorial optimization problem; the second was a shift planning and scheduling problem aimed at determining the structure of the permanent workforce at USPS mail processing and distribution centers.

Chapter 2 described how stochastic models can be derived from deterministic models which are often simplifications obtained by replacing random parameters with their mean values. Standard procedures for comparing solutions obtained from stochastic and deterministic models were also discussed. In the two applications, the value of the stochastic solution (VSS) was reasonably high, meaning that the stochastic model provided some advantage over its deterministic counterpart.

For the generalized assignment problem, the resource capacities and processing times were regarded as (discrete) random variables and random parameters were used for the analysis. The computational results showed improvement between the stochastic solution and the EEV solution that ranged from 5% up to 110%. In the Dantzig-Wolfe decomposition approach, the pricing problems were integer problems. While the computational results were satisfactory, algorithmic performance could certainly be improved by finding a more efficient way to solve them. The literature reports that the deterministic GAP can be solved by CPLEX in reasonable time when the problem size is modest. However, the computational time for the stochastic GAP by CPLEX increased significantly with the same number of jobs and agents.

Personnel scheduling in the service industry presents significant modeling and computational challenges for staff planners. In this study, we have proposed several models that can be used to help size a permanent workforce at facilities that operate 24 hours a day and experience wide fluctuations in demand. For a three-scenario problem, we were able to find high quality solutions to the corresponding SIPs in a reasonable amount of time. This is critical when parametric analysis is an integral component of the planning process.

In Chapter 3, a branch-and-price algorithm was developed for the two applications. The GAP proved much easier to solve with this algorithm than the staff scheduling problem. When the master problem was formulated with nonanticipativity constraints, the column generation component of algorithm bogged down due to the large number of constraints in the formulation. Column generation implementations for shift scheduling problem that use *cover-*

ing constraints in the master problem rather than nonanticipativity constraints have been the most successful. For a stochastic lot-sizing problem, Lulli and Sen [48] included nonanticipativity constraints in the master problem but its row dimension was less than 200, much smaller than the problems considered in this dissertation. The column generation algorithm implemented for the shift planning and scheduling problem was effective for only small problem instances.

As expected, column generation provided tighter LP bounds than conventional LP relaxation. To find integer solutions, a branch-and-bound procedure was developed as part of the branch-and-price algorithm. Large instances, however, failed to converge in several hours suggesting that a better balance is needed between computational efficiency and solution quality. In this regard, a heuristic that may be worth pursuing is to terminate column generation prior to obtaining zero objection function values for all subproblems. Of course, when a node in the search tree is fathomed prematurely, the optimal solution may be overlooked. For large instances, though, this might be acceptable if a fathoming strategy can be developed that generally results in near-optimal solutions.

Future research can be divided into two parts—identifying more effective decompositions and developing more efficient algorithms. If we can formulate a column generation model for the shift planning and scheduling problem with a covering constraint, then column generation may prove more practical. For the GAP, another approach such as a dynamic programming for solving the pricing problems might improve its performance.

In this research, one version of the GAP was studied, leaving open the opportunity to explore several others. In the GAP investigated here, the second stage variables denoted by y measured the amount of unmet capacity and were treated as continuous. Capacity violations were penalized in the objective function. Other possibilities would be to penalize the number of infeasible scenarios instead or to allow the first stage assignments, x, to be cancelled in the second stage by paying a penalty. Those problems were studied by Tokas et al. [66] but were not solved with column generation.

Bibliography

[1] Albareda-Sambola, M., M. H. van der Vlerk, and E. Fernández (2002). Exact solutions to a class of stochastic generalized assignment problems. *Stochastic Programming E-Print Series 2002-4*, `http://www.speps.org`.

[2] Aykin, T. (1996). Optimal shift scheduling with multiple break windows. *Management Science 42*(4), 591–602.

[3] Bailey, J. (1985). Integrated days off and shift personnel scheduling. *Computers & Industrial Engineering 9*(4), 395–404.

[4] Balachandran, V. (1976). An integer generalized transportation model for optimal job assignment in computer networks. *Operations Research 24*(4), 742–759.

[5] Bard, J. F. (2004). Selecting the appropriate input data set when configuring a permanent workforce. *Computers & Industrial Engineering 47*(4), 371–389.

[6] Bard, J. F., C. Binici, and A. H. deSilva (2002). Staff scheduling at the united states postal service. *Computers & Operations Research 30*(5), 745–771.

[7] Bard, J. F. and S. Rojanasoonthon (2006). A branch-and-price algorithm for parallel machine scheduling with time windows and job priorities. *Naval Research Logistics 53*(1), 24–44.

[8] Barnhart, C., E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations Research 46*(3), 316–329.

[9] Beale, E. M. L. (1955). On minimizing a convex function subject to linear inequalities. *Journal of the Royal Statistical Society 17*(2), 173–184.

[10] Beaumont, N. (1997). Using mixed integer programming to design employee rosters. *Journal of the Operational Research Society 48*(6), 585–590.

[11] Bechtold, S. E. and L. W. Jacobs (1990). Implicit modeling of flexible break assignments in optimal shift scheduling. *Management Science 36*(11), 1339–1351.

[12] Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik 4*(3), 238–252.

[13] Bertsimas, D. and J. N. Tsitsiklis (1997). *Introduction to Linear Optimization*. Belmont, MA: Arthena Scientific.

[14] Birge, J. R. and F. Louveaux (1997). *Introduction to Stochastic Programming*. New York: Springer.

[15] Brusco, M. J. and L. W. Jacobs (1998). Personal tour scheduling when starting time restrictions are present. *Management Science 44*(4), 534–547.

[16] Burns, R. N. and M. W. Carter (1985). Work force size and single shift schedules with variable demands. *Management Science 31*(5), 599–607.

[17] Carøe, C. C. and R. Schultz (1999). Dual decomposition in stochastic integer programming. *Operations Research Letters 24*(1–2), 37–45.

[18] Carøe, C. C. and J. Tind (1998). L-shaped decomposition of two-stage stochastic programs with integer recourse. *Mathematical Programming 83*(3), 407–424.

[19] Dantzig, G. B. (1954). A comment on edie's traffic delays at toll booths. *Operations Research 3*(3), 339–341.

[20] Dantzig, G. B. (1955). Linear programming under uncertainty. *Management Science 1*(3), 197–206.

[21] Dantzig, G. B. and P. Wolfe (1960). Decomposition principles for linear programs. *Operations Research 8*(3), 101–111.

[22] Dentcheva, D., R. Gollmer, A. Möller, W. Römisch, and R. Schultz (1999). Solving the unit commitment problem in power generation by primal and dual methods. In *Proceedings of the 13th Power Systems Computation Conference*, Trondheim, pp. 724–730.

[23] Desrochers, M. and F. Soumis (1989). A column generation approach to the urban transit crew scheduling problem. *Transportation Science 23*(1), 1–13.

[24] Desrosiers, J., F. Soumis, and M. Desrochiers (1984). Routing with time windows by column generation. *Networks 14*(4), 545–565.

[25] du Merle, O., D. Villeneuve, J. Desrosiers, and P. Hansen (1999). Stabilized column generation. *Discrete Mathematics 194*(1–3), 229–237.

[26] Escudero, L. F., P. V. Kamesam, A. J. King, and R. J.-B. Wets (1993). Production planning via scenario modeling. *Annals of Operations Research 43*(6), 311–335.

[27] Fisher, M. L., R. Jaikumar, and L. N. van Wassenhove (1986). A multiplier adjustment method for the generalized assignment problem. *Management Science 32*(9), 1095–1103.

[28] Gavish, B. and H. Pirkul (1991). Algorithms for the multi-resource generalized assignment problem. *Management Science 37*(6), 695–713.

[29] Geoffrion, A. M. (1972). Generalized benders decomposition. *Journal of Optimization Theory and Applications 10*(4), 237–260.

[30] Gilmore, P. C. and R. E. Gomory (1963). A linear programming approach to the cutting stock problem–part II. *Operations Research 11*(6), 863–888.

[31] Goffin, J.-L. and J.-P. Vial (2002). Convex nondifferentiable optimization: A survey focused on the analytic center cutting plan method. *Optimization Methods and Software 17*(5), 805–867.

[32] Gollmer, R., M. P. Nowak, W. Römisch, and R. Schultz (2000). Unit commitment in power generation - a basic model and some extensions. *Annals of Operations Research 96*(1), 167–189.

[33] Guignard, M. and M. B. Rosenwein (1989). An improved dual based algorithm for the generalized assignment problem. *Operations Research 37*(4), 658–663.

[34] Higle, J. L. and S. Sen (1996). *Stochastic Decomposition: A Statistical Method for Large Scale Stochastic Linear Programming*. Dordrecht, Netherlands: Kluwer Academic Publisher.

[35] IBM Watson Research Center (2003). *COIN-OR: Common Optimization INterface for Operations Research*. `http://www.coin-or.org`: IBM Watson Research Center.

[36] Infanger, G. (1994). *Planning under Uncertainty: Solving Large-Scale Stochastic Linear Programs*. Massachusetts: Boyd and Fraser.

[37] Jarrah, A. I. Z., J. F. Bard, and A. H. deSilva (1994). Solving large-scale tour scheduling problems. *Management Science 40*(9), 1124–1145.

[38] Johnson, E. L. (1989). Modeling and strong linear programs for mixed-integer programming. In S. W. Wallace (Ed.), *Algorithms and Model For-*

mulations in Mathematical Programming, Volume 51 of *NATO ASI Series*, pp. 1–41.

[39] Johnson, R. A. and D. W. Wichern (1992). *Applied Multivariate Statistical Analysis*. New York: Prentice Hall.

[40] Kall, P. and S. W. Wallace (1994). *Stochastic Programming*. Chichester, UK: John Wiley & Sons.

[41] Kiwiel, K. C. (1993). *User's Guide for NOA 3.0: A Fortran Package for Convex Nondifferentiable Optimization*. Warsaw, Poland: Systems Research Institute.

[42] Klein Haneveld, W. K., L. Stougie, and M. H. van der Vlerk (1995). On the convex hull of the simple integer recourse objective function. *Annals of Operations Research 56*(1), 209–224.

[43] Laprote, G. and F. V. Louveaux (1993). The integer L-shaped method for stochastic integer programs with complete recourse. *Operations Reserach Letters 13*(3), 133–142.

[44] Laprote, G., F. V. Louveaux, and L. van Hamme (1994). Exact solution to a location problem with stochastic demands. *Transportation Science 28*(2), 95–103.

[45] Lau, H. C. (1996). On the complexity of manpower shift scheduling. *Computers & Operations Research 23*(1), 93–102.

[46] Li, D., X. L. Sun, M. P. Biswal, and F. Gao (2001). Convexification, concavification and monotonization in global optimization. *Annals of Operations Research 105*(1), 213–226.

[47] Lübbecke, M. E. and J. Desrosiers (2005). Selected topics in column generation. *Operations Research 53*(6), 1007–1023.

[48] Lulli, G. and S. Sen (2004a). A branch-and-price algorithm for multistage stochastic integer programming with application to stochastic batch-sizing problems. *Management Science 50*(6), 786–796.

[49] Lulli, G. and S. Sen (2004b). A branch and price algorithm for multistage stochastic integer programming with application to stochastic batch-sizing problems. *Management Science 50*(6), 786–796.

[50] Marsten, R. E., W. W. Hogan, and J. W. Blankenship (1975). The boxstep method for large-scale optimization. *Operations Research 23*(3), 389–405.

[51] Martello, S. and P. Toth (1981). An algorithm for the generalized assignment problem. In J. P. Brans (Ed.), *Operations Research '81*, pp. 589–603. Amsterdam: North-Holland.

[52] Martello, S. and P. Toth (1990). *Knapsack problems: Algorithms and computer implementations*. Chichester, UK: John Wiley & Sons.

[53] McGill, J. I. and G. J. van Ryzin (1999). Revenue management: Research overview and prospects. *Transportation Science 33*(2), 233–256.

[54] Narciso, M. G. and L. A. N. Lorena (1999). Lagrangean/surrogate relaxation for generalized assignment problems. *European Journal of Operational Research 114*(1), 165–177.

[55] Prékopa, A. (1995). *Stochastic Programming.* Dordrecht, Netherlands: Kluwer Academic Publisher.

[56] Rosa, C. H. and A. Ruszczyński (1996). On augmented lagrangian decomposition methods for multistage stochastic programs. *Annals of Operations Research 64*(1), 289–309.

[57] Ross, G. T. and R. M. Soland (1975). A branch-and-bound algorithm for the generalized assignment problem. *Mathematical Programming 8*(1), 91–103.

[58] Ross, G. T. and R. M. Soland (1977). Modeling facility location problem as generalized assignment problems. *Management Science 24*(3), 345–357.

[59] Rousseau, L. M., M. Gendreau, and D. Feillet. Interior point stabilization for column generation. Working paper, Universite d'Avignon, Cedex 9, France.

[60] SAS Institute Inc. (2003). *SAS/STAT User's Guide, Release 8.1.* Cary, NC: SAS Institute Inc.

[61] Savelsbergh, M. (1997). A branch-and-price algorithm for the generalized assignment problem. *Operations Research 45*(6), 831–841.

[62] Segal, M. (1974). The operator scheduling problem: A network flow approach. *Operations Research 22*(4), 803–824.

[63] Silva, E. F. and R. K. Wood (2006). Solving a class of stochastic mixed-integer programs with branch and price. *Mathematical Programming*, To appear.

[64] Spoerl, D. R. and R. K. Wood (2006). A stochastic generalized assignment problem. Working paper, Naval Postgraduate School, Monterey, CA.

[65] Takriti, S., J. R. Birge, and E. Long (1996). A stochastic model for the unit commitment problem. *IEEE Transactions on Power Systems 11*(3), 1497–1508.

[66] Tokas, B., J. W. Yen, and Z. B. Zabinsky (2004). A stochastic programming approach to resource-constrained assignment problems. *Stochastic Programming E-Print Series 2004-13*, `http://www.speps.org`.

[67] Trick, M. (1992). A linear relaxation heuristic for the generalized assignment problem. *Naval Research Logistics 39*(1), 137–151.

[68] Van der Vlerk, M. H. (1997). Convex approximations for stochastic programs with simple integer recourse. In *Ten years LNMB*, Volume 122 of *CWI Tract*, pp. 357–365. Amsterdam: Math. Centrum Centrum Wisk. Inform.

[69] Van der Vlerk, M. H. (2002). Convex approximations for complete integer recourse models. *Mathematical Programming*, To appear.

[70] Van Slyke, R. M. and R. J.-B. Wets (1969). L-shaped linear programs with application to optimal control and stochastic programming. *SIAM Journal of Applied Mathematics 17*(4), 638–663.

[71] Van Slyke, R. M. and Y. Young (2000). Finite horizon stochastic knapsacks with applications to yield management. *Operations Research 48*(1), 155–172.

[72] Vanderbeck, F. and L. A. Wolsey (1996). An exact algorithm for IP column generation. *Operations Research Letter 19*(4), 151–159.

[73] Wentges, P. (1997). Weighted Dantzig-Wolfe decomposition for linear mixed-integer programming. *International Transactions in Operational Research 4*(2), 151–162.

[74] Wollmer, R. D. (1980). Two-stage linear programming under uncertainty with 0-1 integer first stage variables. *Mathematical Programming 19*(1), 279–288.

[75] Zhang, X. (2003). *Equipment Scheduling at Mail Processing and Distribution Centers*. Ph. D. thesis, Graduate Program in Opeartions Research & Industrial Engineering, The University of University, Austin.

Vita

Yong Min Wang was born in Chonju, Chullanam-do, South Korea on February 10, 1967 to San-Seog Wang and Sunok Sung. He received the Bachelor of Science degree in Industrial Engineering with second-class honors from the Hanyang University, Seoul, South Korea in 1989. After earning the Master of Science degree with a 4.0 GPA in 1991, he worked for Samsung Electronics Co. Ltd., as a computer programmer, market analyst and product designer. He left Samsung in 1998 to resume his studies at Columbia University, NY. He received his second Master of Science degree in Operations Research from Columbia University in 2000 and continued on to pursue a Ph.D. in the Graduate Program in Operations Research at the University of Texas at Austin. Yong Min has been married to Suk-Hwa Yun since May 7, 1994.

Permanent address: 2501 Lake Austin Blvd #B202
Austin, Texas 78703

This dissertation was typeset with LaTeX† by the author.

†LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.